رسانهٔ انتقال می‌تواند **هدایت‌پذیر** (guided) باشد یا **هدایت‌ناپذیر** (unguided) . مهمترین رسانه‌های هدایت‌پذیر عبارتند از: زوج-تابیده، کابل کواکسیال، و فیبر نوری. رسانه‌های هدایت‌ناپذیر نیز عبارتند از: امواج رادیویی، مایکروبو، مادون‌قرمز، و لیزر. یکی از رسانه‌های انتقال رو به رشد ماهواره‌های مخابراتی (بویژه سیستمهای مدار پائین - LEO) هستند.

سیستم تلفن یکی از کلیدی‌ترین اجزای شبکه‌های گسترده (WAN) است، که مهمترین عناصر آن عبارتند از: مدارهای پایانی (local loop) ، ترانک‌ها (trunk) ، و سوئیچ‌ها (switch) . مدارهای پایانی مدارهای آنالوگ زوج-تابیده هستند، که برای انتقال داده‌های دیجیتال روی آنها باید از **مودم** (modem) استفاده کرد. ADSL با استفاده از تکنیکهای مدولاسیون و تقسیم مدار پایانی به کانالهای مجازی می‌تواند به سرعت 50 Mbps برسد. **مدارهای پایانی بیسیم** (WLL) یکی از تکنولوژیهای جدیدیست که در آینده از آن (بویژه از LMDS) بیشتر خواهید شنید.

ترانک‌های شبکهٔ تلفن دیجیتالی هستند، و از تکنیکهای مالتی‌پلکس (شامل FDM ، TDM و WDM) در آنها استفاده می‌شود. تکنیکهای سوئیچینگ نیز بر دو نوع **سوئیچینگ مداری** (circuit switching) و **سوئیچینگ بسته‌ای** (packet switching) است، که هر دو اهمیت زیادی دارند.

برای کاربردهایی که تحرک زیادی دارند، سیستم تلفن ثابت چندان مناسب نیست. تلفنهای همراه امروزه بطور گسترده‌ای برای ارتباطات صدا مورد استفاده قرار می‌گیرند، و در آیندهٔ نزدیک ترافیک داده نیز در آنها بحد قابل‌ملاحظه‌ای خواهد رسید. نسل اول تلفنهای همراه آنالوگ بود، که عمدتاً به سیستمهای AMPS متکی بود. نسل دوم تلفن همراه دیجیتال است، که در آن از سیستمهای D-AMPS ، GSM و CDMA استفاده می‌شود. نسل سوم تلفنهای همراه نیز دیجیتال خواهد بود، و در آن CDMA **پهن‌باند** استفاده خواهد شد.

یکی از سیستمهایی که می‌توان از آن برای کاربردهای شبکه نیز بهره گرفت، تلویزیون کابلی (که از تلویزیون با آنتن مرکزی به سیستمهای آمیخته فیبر-کواکس تکامل یافته) است. این سیستم پهنای باند بالقوه زیادی دارد، ولی پهنای باند واقعی آن به تعداد کاربران فعال (و اینکه مشغول چه کاری هستند) بستگی دارد.

مسائل

۱. ضرایب فوریه تابع $f(t) = t$ را محاسبه کنید $(0 \le t \le 1)$.

۲. هر 1 msec از یک کانال بدون نویز 4-kHz نمونه‌برداری می‌شود. حداکثر نرخ دادهٔ این کانال چقدر است؟

۳. کانالهای تلویزیونی 6 MHz پهنا دارند. اگر از یک سیگنال دیجیتال چهارسطحی استفاده کنیم، چند bit/sec می‌توان در این کانال مخابره کرد؟ فرض کنید کانال بدون نویز است.

٤. اگر یک سیگنال باینری در کانالی 3-kHz که نسبت سیگنال به نویز آن 20 dB است، مخابره شود، حداکثر نرخ دادهٔ قابل دستیابی چقدر است؟

۵. برای آن که بتوان کاریر T1 را روی یک خط 50-kHz قرار داد، نسبت سیگنال به نویز چقدر باید باشد؟

۶. فرق ستارهٔ غیرفعال و تکرارکنندهٔ فعال در یک شبکهٔ فیبر نوری چیست؟

۷. پهنای باند موجود در طیفی به پهنای 0.1 μm در طول موج 1 μm چقدر است؟

۸. می‌خواهیم یکسری تصاویر کامپیوتری اسکن‌شده را روی یک رشته فیبر نوری بفرستیم. وضوح هر تصویر 640 × 480 پیکسل، و هر پیکسل ۲۴ بیت است. تصاویر با سرعت ۶۰ صفحه بر ثانیه اسکن می‌شوند. پهنای باند موردنیاز چقدر است؟ اگر از باند 1.30 μm استفاده کنیم، به چند میکرون از طول موج نیاز داریم؟

۹. آیا قضیهٔ نایکوئیست برای فیبرهای نوری هم صادق است، یا فقط برای کابلهای مسی کاربرد دارد؟

۱۰. در شکل ۲-۶، باند سمت چپ از بقیهٔ باندها باریکتر است. چرا؟

۱۱. یک آنتن زمانی بهترین بهره را دارد که قطر آن معادل طول موج امواج رادیویی باشد. قطر قابل قبول آنتنها بین ۱ تا ۵ متر است. این قطر معادل کدام طیف فرکانسی است؟

۱۲. محوشدگی چندمسیره زمانی به حداکثر می‌رسد که دو موج با اختلاف فاز ۱۸۰ درجه وارد گیرنده شوند. برای به حداکثر رسیدن محوشدگی چندمسیره در یک لینک مایکروویو 1-GHz بطول 50-km این مقدار چقدر باید باشد؟

۱۳. پرتو لیزری بقطر 1 mm روی آشکارسازی بقطر 1 mm که روی پشت‌بامی در فاصلهٔ 100 m قرار دارد، نشانه گرفته شده است. حداکثر انحراف زاویه‌ای (بر حسب درجه) چقدر باید باشد، تا پرتو لیزری هدف را گم نکند؟

۱۴. ۶۶ ماهوارهٔ پروژهٔ ایریدیوم به شش کمربند بدور زمین تقسیم شده‌اند. در ارتفاعی که این ماهواره‌ها قرار دارند، دورهٔ گردش مداری ۹۰ دقیقه است. متوسط زمان پاس‌کاری یک فرستندهٔ زمینی بین دو ماهواره چقدر است؟

۱۵. ماهواره‌ای در مدار زمین‌ثابت با صفحهٔ استوای زمین زاویهٔ ϕ می‌سازد. آیا برای فردی که روی زمین در مدار ϕ درجهٔ شمالی ایستاده، این ماهواره در آسمان ثابت بنظر می‌رسد؟ اگر نه، حرکت آنرا توضیح دهید.

۱۶. قبل از سال ۱۹۸۴ (وقتی هر ایستگاه پایانی با کُد سه رقمی ناحیه و سه رقم اول شماره تلفن مشخص می‌شد) در سیستم تلفن چند کُد ایستگاه پایانی می‌توانست وجود داشته باشد؟ کُد ناحیه با عددی بین 2 تا 9 شروع می‌شد، رقم دوم می‌توانست 0 یا 1 باشد، و رقم سوم محدودیتی نداشت. دو رقم اول کُد محلی نیز بایستی بین 2 تا 9 ، و رقم سوم می‌توانست هر عددی باشد.

۱۷. فقط با اطلاعاتی که در اینجا بدست آوردید، آیا می‌توانید بگوئید حداکثر شماره تلفنهایی که (بدون تغییر در روش شماره‌گذاری یا اضافه کردن تجهیزات) می‌توان در ایالات متحده نصب کرد، چه تعداد است؟ آیا این تعداد شماره قابل دستیابی است؟ هر دستگاه فکس یا کامپیوتر را نیز یک تلفن در نظر بگیرید، و فرض کنید هر مشترک فقط یک دستگاه تلفن دارد.

۱۸. یک سیستم سادهٔ تلفن را که در آن دو ایستگاه پایانی و یک ایستگاه بین‌شهری بوسیلهٔ خطوط دو-طرفهٔ همزمان 1-MHz به هم متصل شده‌اند، در نظر بگیرید. هر دستگاه تلفن بطور متوسط در هر روز کاری ۸ ساعته ۴ تماس برقرار می‌کند، و زمان متوسط هر تماس ۶ دقیقه است. ده درصد تماسها راه دور هستند (یعنی از ایستگاه بین‌شهری رد می‌شوند). حداکثر تعداد شماره‌هایی که هر ایستگاه پایانی می‌تواند پشتیبانی کند، چقدر است؟ مدارها را 4-kHz در نظر بگیرید.

۱۹. یک شرکت تلفن منطقه‌ای ۱۰ میلیون مشترک دارد، که بوسیلهٔ زوج-تابیده به ایستگاه مرکزی متصل شده‌اند. متوسط طول مدارهای پایانی ۱۰ کیلومتر است. ارزش مس موجود در مدارهای پایانی این سیستم چقدر است؟ سطح مقطع سیمها را دایره‌ای بقطر 1 mm ، چگالی مس را $9.0\ gr/cm^3$ و ارزش هر کیلوگرم مس را 3 دلار در نظر بگیرید.

۲۰. یک خط لولهٔ نفت سیستمی یکطرفه است، یا دو-طرفهٔ ناهمزمان، یا دو-طرفهٔ همزمان، یا هیچکدام؟

۲۱. قیمت میکروپروسسورهای سریع آنقدر کاهش یافته، که می‌توان در هر دستگاه مودم یک میکروپروسسور قرار دارد. این کار چه تأثیری روی مقابله با خطاهای خطوط تلفن دارد؟

۲۲. دیاگرام فلکی شکل ۲-۲۵ چهار نقطهٔ داده در مختصات (1, 1) ، (1, -1) ، (-1, 1) و (-1, -1) دارد. مودمی با این پارامترها در 1200 baud به چه سرعتی (bps) می‌تواند دست یابد؟

۲۳. مودمی با دیاگرام فلکی شبیه شکل ۲-۲۵ دارای نقاط داده‌ای در مختصات (0, 1) و (0, 2) است. این مودم

از مدولاسیون فاز استفاده می‌کند، یا مدولاسیون دامنه؟

۲۴. در یک دیاگرام فلکی تمام نقاط روی دایره‌ای به مرکز مبدأ مختصات واقع شده‌اند. مدولاسیون این مودم چیست؟

۲۵. یک مودم دو-طرفهٔ همزمان QAM-64 از چند فرکانس استفاده می‌کند؟

۲۶. در یک سیستم ADSL که از DMT استفاده می‌کند، 3/4 کانالهای موجود به لینک دریافت اختصاص داده شده است. هر کانال از مدولاسیون QAM-64 استفاده می‌کند. ظرفیت لینک دریافت چقدر است؟

۲۷. در سیستم LMDS چهار قطاعی شکل ۲-۳۰، هر قطاع یک کانال اختصاصی 36-Mbps دارد. طبق تئوری صف، اگر 50% کانال پُر باشد، زمان انتظار در صف معادل زمان بار شدن است. در چنین شرایطی، بار شدن یک صفحهٔ وب 5-KB چقدر طول خواهد کشید؟ بار شدن این صفحه روی یک خط ADSL با سرعت 1 Mbps چقدر طول می‌کشد؟ با یک مودم 56-kbps چقدر؟

۲۸. ده سیگنال، که هر کدام به پهنای باند 4000 Hz نیاز دارند، با استفاده از FDM روی یک کانال مالتی‌پلکس شده‌اند. حداقل پهنای باند موردنیاز این کانال چقدر است؟ پهنای باندهای محافظ را 400 Hz در نظر بگیرید.

۲۹. چرا زمان نمونه‌برداری PCM در 125 μsec ثابت شده است؟

۳۰. درصد سرآیند یک کاربر T1 (درصدی از 1.544 Mbps که بکار داده‌های کاربر نمی‌آید) چقدر است؟

۳۱. حداکثر نرخ دادهٔ یک کانال بدون‌نویز 4-kHz را با استفاده از تکنیکهای زیر مقایسه کنید:

(الف) کُدگذاری آنالوگ (مثلاً، QPSK) با 2 bits/sample .

(ب) سیستم T1 PCM .

۳۲. اگر یک سیستم T1 دچار لغزش شود، برای سنکرون شدن مجدد از اولین بیت هر فریم استفاده می‌کند. برای سنکرون شدن مجدد با احتمال خطای 0.001 ، چند فریم باید بررسی شود؟

۳۳. فرق بخش دمودلاتور یک مودم با بخش دکودر یک کُدک چیست (و آیا اساساً فرقی دارند)؟ توجه داشته باشید که هر دوی اینها سیگنالهای آنالوگ را به دیجیتال تبدیل می‌کنند.

۳۴. سیگنالی بصورت دیجیتال روی یک کانال بدون‌نویز 4-kHz (با یک نمونه در هر 125 μsec) فرستاده می‌شود. با هر یک از روشهای کُدگذاری زیر چند بیت در ثانیه ارسال می‌شود:

(الف) استاندارد CCITT 2.048 Mbps .

(ب) سیستم DPCM با مقدار نسبی سیگنال 4-bit .

(ج) مدولاسیون دلتا.

۳۵. یک سیگنال سینوسی کامل با دامنهٔ A با استفاده از مدولاسیون دلتا (با x samples/sec) کُد شده است. خروجی 1+ معادل $A/8$+ تغییر در سیگنال ورودی، و خروجی 1- معادل $A/8$- تغییر در سیگنال ورودی است. بیشترین فرکانسی که این سیستم می‌تواند بدون خطای تجمعی تعقیب کند، چقدر است؟

۳۶. ساعتهای SONET خطایی معادل 1 در 10^9 دارند. چه مدت طول می‌کشد، تا این اختلاف باندازهٔ 1 بیت شود؟ عوارض جانبی این پدیده چیست؟

۳۷. در شکل ۲-۳۷، نرخ دادهٔ کاربر OC-3 از 148.608 Mbps شروع شده است. نشان دهید این عدد چگونه از پارامترهای SONET OC-3 بدست آمده است.

۳۸. سیستم SONET برای انطباق با سرعتهای پائین‌تر از STS-1 از روشی بنام **انشعابات مجازی** (Virtual Tributaries - VT) استفاده می‌کند. یک VT عبارتست از یک بار جزئی، که می‌توان آنرا بهمراه بارهای

جزئی دیگر در یک فریم STS-1 قرار داد. VT1.5 از ۳ ستون، VT2 از ۴ ستون، VT3 از ۶ ستون و VT6 از ۱۲ ستون فریم STS-1 استفاده می‌کنند. کدام VT با هر یک از سرویسهای زیر منطبق است؟

(الف) سرویس DS-1 (1.544 Mbps) .

(ب) سرویس European CEPT-1 (2.048 Mbps) .

(ج) سرویس DS-2 (6.312 Mbps) .

۳۹. تفاوت بنیادی سوئیچینگ مداری با سوئیچینگ بسته‌ای چیست؟

۴۰. پهنای باند کاربر یک اتصال OC-12c چقدر است؟

۴۱. سه شبکهٔ سوئیچینگ بسته‌ای هر کدام ۳ گره دارند. شبکهٔ اول دارای توپولوژی ستاره (با سوئیچ مرکزی) است، شبکهٔ دوم حلقه (دوطرفه) است، و شبکهٔ سوم اتصالات کامل داخلی دارد (یعنی هر گره مستقیماً به تمام گره‌های دیگر متصل است). بهترین، بدترین و متوسط پرش (hop) در ارتباط از یک نقطه به نقطهٔ دیگر در هر یک از این شبکه‌ها چیست؟

۴۲. زمان تأخیر ارسال یک پیام x-bit در مسیری با k پرش در یک شبکهٔ سوئیچینگ مداری و یک شبکهٔ سوئیچینگ بسته‌ای (با بار کم) را با یکدیگر مقایسه کنید. زمان برقرای مدار را s ثانیه، زمان تأخیر در هر پرش را d ثانیه، اندازهٔ هر بسته را p بیت، و نرخ انتقال داده را b pbs در نظر بگیرید. در چه شرایطی تأخیر ارسال شبکهٔ سوئیچینگ بسته‌ای کمتر است؟

۴۳. فرض کنید می‌خواهیم x بیت اطلاعات را در یک شبکهٔ سوئیچینگ بسته‌ای با k پرش، بصورت بسته‌هایی با p بیت داده و h بیت سرآیند (با این فرض که $x >> p + h$) منتقل کنیم. نرخ انتقال دادهٔ خطوط b bps، و زمان تأخیر انتشار در آنها قابل صرفنظر کردن است. چه مقداری از p تأخیر کلی را به حداقل می‌رساند؟

۴۴. در یک سیستم تلفن همراه با سلولهای شش ضلعی، استفاده از باندهای فرکانسی مشابه در سلولهای مجاور ممنوع است. اگر ۸۴۰ باند فرکانسی داشته باشیم، در هر سلول از چند فرکانس می‌توان استفاده کرد؟

۴۵. طرح کلی سلولهای یک شبکهٔ تلفن همراه بندرت مانند شکل ۲-۴۱ منظم است؛ حتی شکل هر سلول نیز منظم نیست. یک علت برای این وضعیت بیاورید.

۴۶. برای پوشش دادن به شهری با مساحت 120 km^2 ، به چه تعداد سلول PCS با قطر 100 m نیاز داریم؟ (تخمین بزنید.)

۴۷. گاهی هنگام عبور یک کاربر تلفن همراه از سلولی به سلول دیگر، با وجود اینکه تمام فرستنده‌ها و گیرنده‌ها بخوبی کار می‌کنند، ارتباط ناگهان قطع می‌شود. چرا؟

۴۸. کیفیت صدای D-AMPS بسیار پائینتر از GSM است. آیا این بخاطر اجبار D-AMPS در حفظ سازگاری با AMPS است (در حالیکه GSM چنین محدودیتی ندارد)؟ یا علت دیگری دارد؟

۴۹. حداکثر تعداد کاربران همزمان در یک سلول D-AMPS را محاسبه کنید. آیا چنین محاسبه‌ای برای GSM هم امکان دارد؟ علت را توضیح دهید.

۵۰. فرض کنید سه ایستگاه A، B و C در یک سیستم CDMA (با توالیهای چیپ شکل ۲-۴۵ ب) همزمان اقدام به ارسال بیتهای 0 می‌کنند. توالی چیپ حاصله چیست؟

۵۱. در بحث متعامد بودن بردارهای توالی چیپ CDMA، گفتیم که اگر $S \bullet T = 0$، آنگاه $S \bullet \overline{T} = 0$. ثابت کنید.

۵۲. اجازه دهید متعامد بودن بردارهای توالی چیپ CDMA را به روشی دیگر بیان کنیم: هر بیت در یک جفت توالی یا یکسان هستند، یا نیستند. متعامد بودن بردارها را با استفاده از اصطلاحات یکسان بودن و یکسان نبودن توضیح دهید.

۵۳. یک گیرندهٔ CDMA توالی (1+ 1+ 3- 1- 3+ 1+ 3- 1+ 1-) را دریافت می‌کند. با فرض توالیهای چیپ شکل ۲-۴۵ (ب)، تعیین کنید کدام ایستگاه‌ها، چه بیت‌هایی را ارسال کرده‌اند؟

۵۴. شبکهٔ تلفن در بخش انتهایی دارای توپولوژی ستاره است، که در آن تمام انشعابات به ایستگاه پایانی ختم می‌شوند. در حالیکه در تلویزیون کابلی، یک کابل مشترک مانند ماری بین مشترکین مختلف خزیده است. فرض کنید در آینده در شبکه‌های کابلی بجای کابلهای مسی از فیبر نوری 10 Gbps استفاده شود. آیا با چنین سیستمی می‌توان مدل شبکهٔ تلفن (یک خط مستقل از هر مشترک به ایستگاه مرکزی) را شبیه‌سازی کرد؟ اگر پاسخ مثبت است، هر فیبر چند کاربر تلفن می‌تواند داشته باشد؟

۵۵. سیستمهای تلویزیون کابلی معمولاً دارای ۱۰۰ کانال تجاری هستند، که بطور متناوب برنامه و آگهی پخش می‌کنند. این سیستم بیشتر شبیه TDM است یا FDM ؟

۵۶. یک شرکت کابلی تصمیم می‌گیرد به ۵۰۰۰ مشترک خود سرویس اینترنت ارائه دهد. این شرکت از کابلهای کواکسیال استفاده می‌کند، که هر کابل می‌تواند تا 100 Mbps روی کانال دریافت از اینترنت ظرفیت داشته باشد. شرکت مزبور برای جذب مشتریان تصمیم می‌گیرد که تا دریافت 2 Mbps را برای هر مشترک تضمین کند. توضیح دهید این شرکت برای رسیدن به هدف فوق چه کاری باید انجام دهد؟

۵۷. با توجه به تخصیص فرکانس نشان داده شده در شکل ۲-۴۸ و اطلاعات داده شده در متن کتاب، یک سیستم کابلی چه مقدار (Mbps) از ظرفیت را به ارسال به اینترنت و چه مقدار را به دریافت از اینترنت اختصاص می‌دهد؟

۵۸. اگر سیستم کابلی کاملاً بیکار باشد، کاربر با چه سرعتی می‌تواند اطلاعات را دریافت کند؟

۵۹. مالتی‌پلکس کردن استریمهای متعدد STS-1 (که به آنها انشعاب گفته می‌شود) نقش مهمی در سیستم SONET بازی می‌کند. یک مالتی‌پلکسر 3:1 سه ورودی STS-1 را در یک خروجی STS-3 مالتی‌پلکس می‌کند. اینکار بصورت بایت به بایت انجام می‌شود، یعنی سه بایت اول خروجی بترتیب بایتهای اول انشعابهای 1 ، 2 و 3 هستند؛ سه بایت دوم خروجی بترتیب بایتهای دوم انشعابهای 1 ، 2 و 3 ؛ و الی آخر. برنامه‌ای بنویسید که این مالتی‌پلکسر 3:1 را شبیه‌سازی کند. برنامهٔ شما باید پنج روال داشته باشد: روال اصلی (که چهار روال دیگر را اجرا می‌کند)، یک روال برای هر یک از انشعابهای STS-1 (مجموعاً سه روال)، و یکی برای مالتی‌پلکسر. هر روال انشعاب یک فریم STS-1 را از فایلی بطول ۸۱۰ بایت خوانده، و این فریمها را (بایت به بایت) به روال مالتی‌پلکسر می‌فرستد. روال مالتی‌پلکسر این بایتها را خوانده، و یک فریم STS-3 را (بایت به بایت) روی خروجی استاندارد (stdout) می‌نویسد. برای ارتباط بین پردازشها از پایپ (pipe) استفاده کنید.

۳ لایه پیوند داده

در این فصل اصول طراحی لایهٔ دوم، **لایهٔ پیوند داده** (data link layer) ، را بررسی خواهیم کرد، و طی آن با الگوریتمهای لازم برای دستیابی به یک ارتباط قابل اطمینان و کارا بین دو کامپیوتر همسایه (در لایهٔ پیوند داده) آشنا خواهید شد. منظور از دو کامپیوتر همسایه، کامپیوترهایی هستند که یک کانال ارتباطی سیم-مانند (کابل کواکسیال، خط تلفن، و یا ارتباط بیسیم) بین آنها برقرار است. خصلت بنیادی یک کانال «سیم-مانند» اینست که بیت‌ها دقیقاً با همان نظمی که فرستاده می‌شوند، در گیرنده دریافت شوند.

شاید در نگاه اول این خصلت آنقدر ساده و ابتدایی بنظر برسد، که فکر کنید چه نیازی به الگوریتم و نرم‌افزار هست: ماشین A بیت‌ها را می‌فرستد، و ماشین B آنها را می‌گیرد. متأسفانه، مسئله بهمین سادگی نیست، چون مدارهای مخابراتی پر از نویز و خطا هستند. علاوه بر آن ظرفیت کانالهای مخابراتی نامحدود نیست، و بین ارسال و دریافت بیت‌ها یک تأخیر زمانی نیز وجود دارد. این محدودیت‌ها تأثیر جدی روی کارایی سیستمهای انتقال داده می‌گذارند. پروتکلهای مخابراتی (که موضوع اصلی این فصل هستند) باید تمام این ملاحظات را در نظر بگیرند.

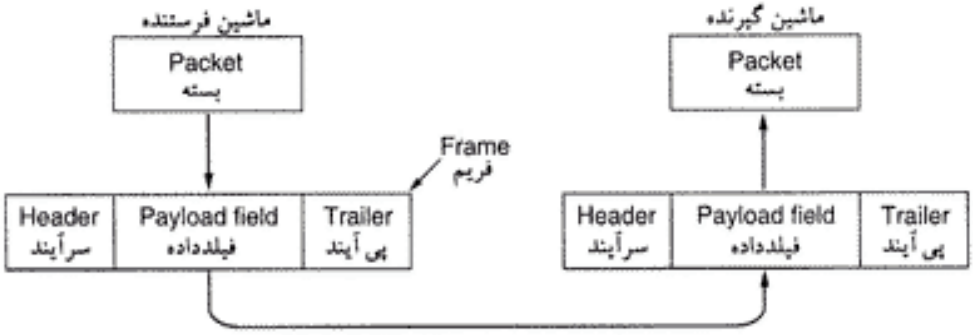
بعد از آشنایی با نکات کلیدی در طراحی لایهٔ پیوند داده، پروتکلهای این لایه را بررسی خواهیم کرد. برای شروع خصلت خطاهای کانالهای مخابراتی، منشأ آنها و نحوهٔ کشف و رفع این خطاها را بررسی کرده، و سپس پروتکلهای لازم برای حل آنها را مورد مطالعه قرار می‌دهیم. در پایان، صحت این مدلها را بررسی کرده، و چند نمونه از پروتکلهای واقعی لایهٔ پیوند داده ارائه خواهیم کرد.

۱-۳ ملاحظات طراحی لایهٔ پیوند داده

لایهٔ پیوند داده وظایف خاصی دارد که باید انجام دهد. این وظایف عبارتند از:

۱. ارائهٔ سرویسهای مشخص به لایهٔ شبکه.
۲. مدیریت خطاهای انتقال.
۳. تنظیم جریان داده‌ها (بگونه‌ایکه گیرنده‌های کُند زیر بمباران فرستنده‌های سریع غرق نشوند).

برای رسیدن به این اهداف، لایهٔ پیوند داده بسته‌های رسیده از لایهٔ شبکه را گرفته و آنها بصورت **فریم** (frame) در می‌آورد. هر فریم سه قسمت دارد: **سرآیند** (header) ، دادهٔ اصلی، و **پی‌آیند** (trailer) ؛ شکل ۳-۱ را ببینید. مدیریت فریمها کلیدی‌ترین وظیفهٔ لایهٔ پیوند داده است، که در بخشهای آینده بتفصیل دربارهٔ آن صحبت خواهیم کرد

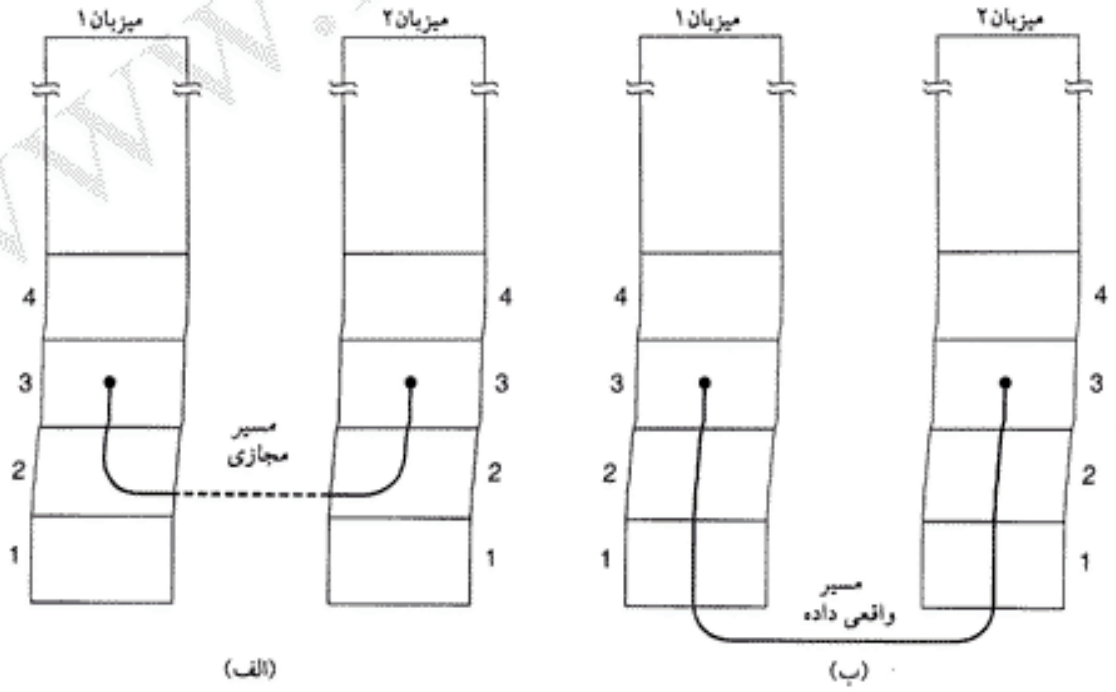


شکل ۳-۱. رابطهٔ بسته و فریم.

با اینکه این فصل منحصراً دربارهٔ لایهٔ پیوند داده و پروتکلهای آن است، اصولی که در اینجا خواهید دید (مانند کنترل خطا و کنترل جریان داده) در لایه‌های دیگر (مانند لایهٔ انتقال) نیز کاربرد دارند. در حقیقت، در بسیاری از شبکه‌ها این کارکردها را فقط در لایه‌های بالاتر (نه در لایهٔ پیوند داده) می‌توانید پیدا کنید. اما صرفنظر از اینکه آنها را کجا می‌توان پیدا کرد، اصول کار یکسان است و اهمیتی ندارد که در این فصل آنها را بررسی کنیم یا فصلهای دیگر. تنها مزیت لایهٔ پیوند داده آنست که این تکنیکها در این لایه ساده‌تر و واضحترند، بنابراین بخوبی می‌توان آنها را مطالعه کرد.

۳-۱-۱ سرویسهایی که به لایهٔ شبکه داده می‌شود

وظیفهٔ لایهٔ پیوند داده ارائهٔ سرویس به لایهٔ شبکه است. مهمترین این وظایف عبارتست از انتقال داده‌ها از لایهٔ شبکهٔ ماشین مبدأ به لایهٔ شبکهٔ ماشین مقصد. در لایهٔ شبکهٔ ماشین مبدأ چیزی هست بنام پروسس، که تعدادی بیت را به لایهٔ پیوند داده می‌دهد تا به مقصدی خاص منتقل کند. وظیفهٔ لایهٔ پیوند داده ماشین مبدأ انتقال این بیت‌ها به ماشین مقصد، و رساندن آنها بدست لایهٔ شبکهٔ مقابل است؛ شکل ۳-۲ (الف) را ببینید. البته مسیری که این بیت‌ها



شکل ۳-۲. (الف) ارتباط مجازی. (ب) ارتباط واقعی.

واقعاً طی می‌کنند، مانند شکل ۳-۲ (ب) است، ولی ساده‌ترست تصور کنیم دو پروسس در لایهٔ پیوند داده آنها را بین خود رد و بدل می‌کنند. بهمین دلیل در این فصل همه جا از مدل شکل ۳-۲ (الف) استفاده خواهیم کرد.

لایهٔ پیوند داده را می‌توان بگونه‌ای طراحی کرد که سرویسهای مختلفی ارائه کند، که این سرویسها از سیستمی به سیستم دیگر متفاوت است. معقولترین این سرویسها عبارتند از:

۱. سرویس غیرمتصل بدون تصدیق دریافت (unacknowledged connectionless).
۲. سرویس غیرمتصل با تصدیق دریافت (acknowledged connectionless).
۳. سرویس اتصال-گرا با تصدیق دریافت (acknowledged connection-oriented).

اجازه دهید این سرویسها را یکی یکی بررسی کنیم.

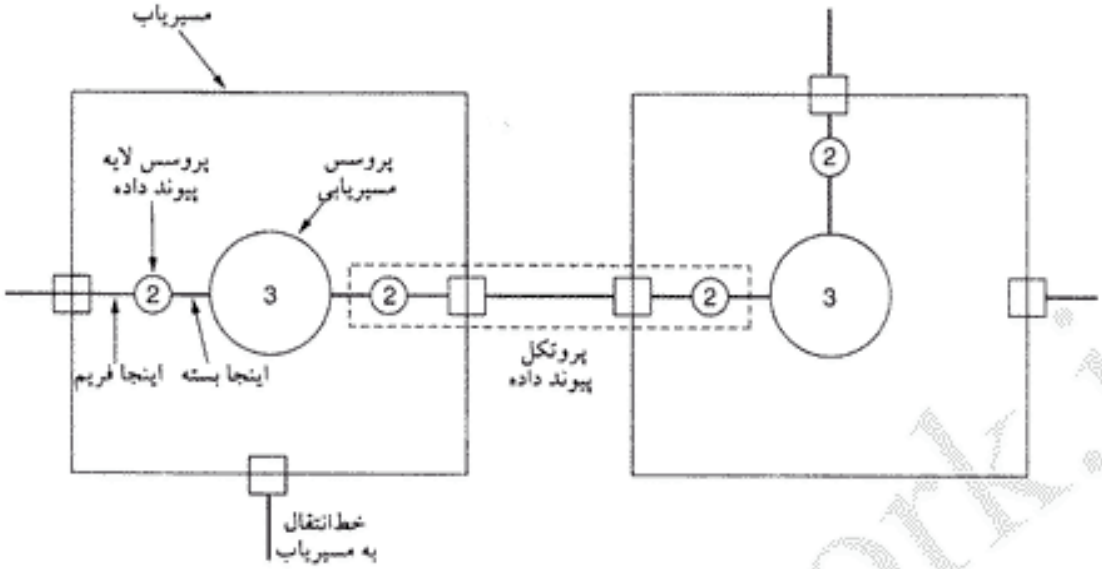
در سرویس **غیرمتصل بدون تصدیق دریافت** ماشین مبدأ فریمهای مستقلی را به ماشین مقصد می‌فرستد، بدون اینکه منتظر تصدیق دریافت آنها از طرف ماشین مقصد بماند. هیچ اتصال منطقی بین دو ماشین برقرار نمی‌شود، پس نیازی به قطع اتصال هم نیست. اگر فریمی در اثر نویز خط از بین برود، هیچ کوششی برای تشخیص این موضوع و مقابله با آن در لایهٔ پیوند داده صورت نمی‌گیرد. این سرویس برای مواقعی مناسب است که نرخ خطا بسیار پائین باشد، و در این حالت مقابله با خطا به لایه‌های بالاتر واگذار می‌شود. این سرویس برای ترافیک زمان-واقعی (مانند سرویس صدا)، که در آن دیر رسیدن بدتر از نرسیدن است، نیز مناسب است. در اغلب LAN ها نیز لایهٔ پیوند داده از سرویسهای غیرمتصل بدون تصدیق دریافت استفاده می‌کند.

سرویس بعدی که قابلیت اعتماد بیشتری دارد، سرویس **غیرمتصل با تصدیق دریافت** است. در این سرویس نیز هیچ اتصال منطقی بین مبدأ و مقصد وجود ندارد، ولی دریافت فریمها از سوی ماشین مقصد تصدیق می‌شود. بدین ترتیب، فرستنده می‌تواند پی ببرد که آیا فریمها بدرستی دریافت شده‌اند یا خیر. اگر فریمی در مدت زمان معین به مقصد نرسد، می‌توان آنرا دوباره ارسال کرد. این سرویس برای کانالهای غیر قابل اعتماد (مانند سیستمهای بیسیم) مناسب است.

لازمست تأکید کنیم که توجه به تصدیق دریافت در لایهٔ پیوند داده فقط برای بهینه‌سازی سیستم است و هیچ الزامی در آن نیست، چون این کار را همیشه می‌توان در لایهٔ شبکه انجام داد. اگر تصدیق دریافت در زمان مشخص از راه نرسد، فرستنده می‌تواند بسته را دوباره ارسال کند. مشکل اینجاست که فریمها معمولاً طول مشخصی دارند، در حالیکه بسته‌ای که لایهٔ شبکه می‌فرستد چنین نیست. اگر بسته‌ای به، مثلاً، ۱۰ فریم شکسته شود، و ۲۰ درصد این فریمها در راه گم شوند، زمان ارسال بسته بسیار طولانی خواهد شد. اما اگر برای هر فریم تصدیق دریافت درخواست شود، این کار سریعتر می‌شود. در کانالهای قابل اعتماد مانند فیبر نوری، بار اضافی چنین پروتکلهای سختگیرانه‌ای در لایهٔ پیوند داده غیر ضروری است، اما در محیطهای ذاتاً پُرنویز مانند بیسیم ارزشش را دارد.

بهترین سرویسی که لایهٔ پیوند داده می‌تواند به لایهٔ شبکه بدهد، سرویس **اتصال-گرا** (connection-oriented) است. در این سرویس قبل از شروع ارسال داده از مبدأ به مقصد، یک اتصال بین آنها برقرار می‌شود. هر فریمی که روی این اتصال فرستاده می‌شود شماره‌گذاری شده است، و لایهٔ پیوند داده دریافت آنها را نیز تضمین می‌کند. همچنین تضمین می‌شود که هر فریم فقط یک بار (و به همان ترتیب ارسال) دریافت شود. اما در سرویسهای غیرمتصل، می‌توان انتظار داشت که بسته‌ای چندین بار ارسال (و در نتیجه چندین بار هم دریافت) شود. سرویس اتصال-گرا استریم قابل اعتمادی از بیت‌ها را در اختیار لایهٔ شبکه می‌گذارد.

ارسال داده‌ها در سرویس اتصال-گرا سه مرحله دارد. در مرحلهٔ اول اتصال برقرار شده، و متغیرهای لازم (برای شمارش فریمها، و اینکه کدام فریمها دریافت شده‌اند و کدامها خیر) ست می‌شوند. در مرحلهٔ دوم، فریمها منتقل می‌شوند. و در مرحلهٔ آخر، اتصال قطع شده و منابع آن (متغیرها و بافرها) آزاد می‌شود.



شکل ۳-۳. محل فعالیت لایهٔ پیوند داده.

اجازه دهید یک مثال بزنیم: یک زیرشبکهٔ WAN متشکل از چند مسیریاب که با خطوط نقطه-به-نقطهٔ تلفن به یکدیگر متصل شده‌اند، را در نظر بگیرید. وقتی یک فریم به مسیریاب می‌رسد، سخت‌افزار (با استفاده از تکنیکهایی که در همین فصل خواهید دید) آنرا از نظر خطا چک می‌کند، و سپس به نرم‌افزار لایهٔ پیوند داده (که می‌تواند روی چیپهای کارت شبکه قرار داشته باشد) تحویل می‌دهد. نرم‌افزار لایهٔ پیوند داده فریم را چک می‌کند تا مطمئن شود همان چیزیست که باید باشد، و اگر چنین بود، قسمت دادهٔ اصلی آنرا به نرم‌افزار مسیریابی می‌دهد نرم‌افزار مسیریابی مسیر خروجی مناسب را تعیین کرده، و بسته را به لایهٔ پیوند داده پس می‌دهد تا ارسال شود (شکل ۳-۳ را ببینید.)

نرم‌افزارهای مسیریابی معمولاً حوصلهٔ بسته‌هایی که مدام گم می‌شوند را ندارند، و دوست دارند بسته‌ها درست و مرتب روی خطوط نقطه-به-نقطه تحویل شوند. این دیگر بر عهدهٔ پروتکل لایهٔ پیوند داده است که خطوط پُرنویز و غیر قابل اعتماد را بصورتی مطمئن (یا نسبتاً مطمئن) در آورد. با اینکه در شکل ۳-۳ نرم‌افزار لایهٔ پیوند داده (در هر مسیریاب) در دو نقطه دیده می‌شود، اما این در واقع یک پروسس واحد است که (به کمک جدول‌ها و ساختمان داده‌های مختلف) تمام کارها را انجام می‌دهد و تمام خطوط را کنترل می‌کند.

۳-۱-۲ فریم‌بندی

لایهٔ پیوند داده به لایهٔ شبکه سرویس می‌دهد، و خود نیز از سرویسهای لایهٔ فیزیکی استفاده می‌کند. چیزی که لایهٔ فیزیکی می‌گیرد، استریمی است از بیت‌ها که باید آنرا به طرف مقابل تحویل دهد. هیچ تضمینی وجود ندارد که این استریم سالم و عاری از خطا به مقصد برسد. تعداد بیتهای رسیده می‌تواند کمتر، مساوی یا بیشتر از بیت‌های ارسال شده باشد، و یا حتی مقدار برخی از آنها تغییر کرده باشد. این بر عهدهٔ لایهٔ پیوند داده است که خطاها را کشف کرده، و در صورت لزوم آنها را برطرف کند.

یکی از روشهای متداول اینست که استریم بیت‌ها در لایهٔ پیوند داده به چند فریم شکسته شده، و برای هر فریم **جمع تطبیقی** (checksum) محاسبه شود. (الگوریتمهای جمع تطبیقی را در همین فصل خواهید دید.) وقتی فریمها به مقصد می‌رسند، جمع تطبیقی آنها مجدداً محاسبه شده و با جمع تطبیقی مبدأ (که به انتهای فریم ضمیمه شده) مقایسه می‌شود. اگر این دو یکی نباشند، لایهٔ پیوند داده متوجه می‌شود که خطایی در فریم رخ داده، و به

سراغ روشهای مقابله با خطا می‌رود (که یکی از این روشها می‌تواند دور انداختن فریم، و درخواست ارسال مجدد آن باشد).

شکستن استریم بیت‌ها به فریم (که به آن **فریم‌بندی** – framing – گفته می‌شود) از آنچه در نگاه اول بنظر می‌رسد، مشکلتر است. یکی از روشهای فریم‌بندی می‌تواند انداختن فاصلهٔ زمانی در نقاطی از استریم بیت‌ها باشد (مانند فاصله انداختن بین کلمات متن). ولی در شبکه‌ها بندرت زمانبندی وجود دارد، و امکان دارد این فاصله‌ها از بین بروند و یا بیشتر شوند.

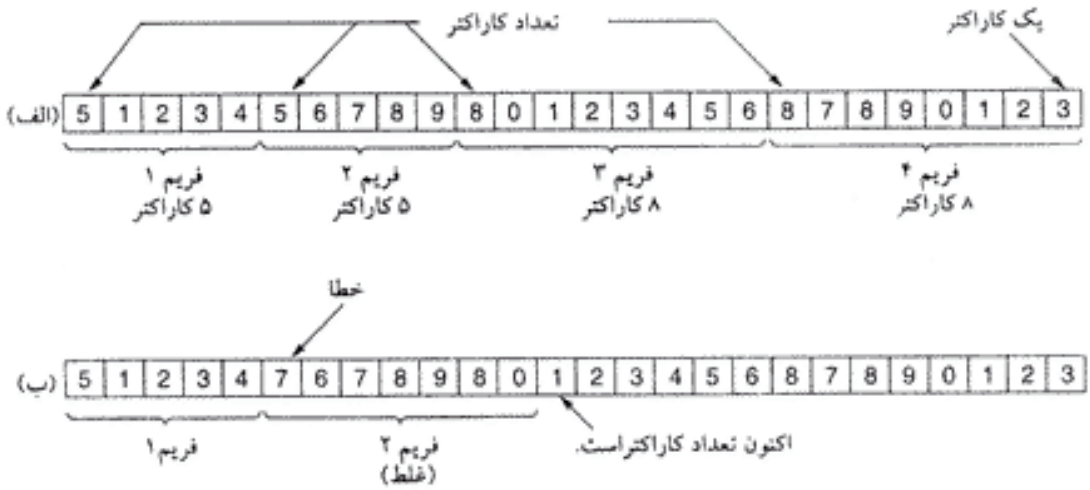
از آنجائیکه تکیه بر زمانبندی برای تعیین ابتدا و انتهای فریمها بسیار خطرناک است، روشهای دیگری برای اینکار ابداع شده، که در این قسمت با چهار تا از آنها آشنا می‌شوید:

۱. **شمارش کاراکترها.**
۲. **بایت‌های پرچم، با لاگذاری بایت.**
۳. **پرچمهای شروع و پایان، با لاگذاری بیت.**
٤. **حالتهای غیرمجاز کُدگذاری لایهٔ فیزیکی.**

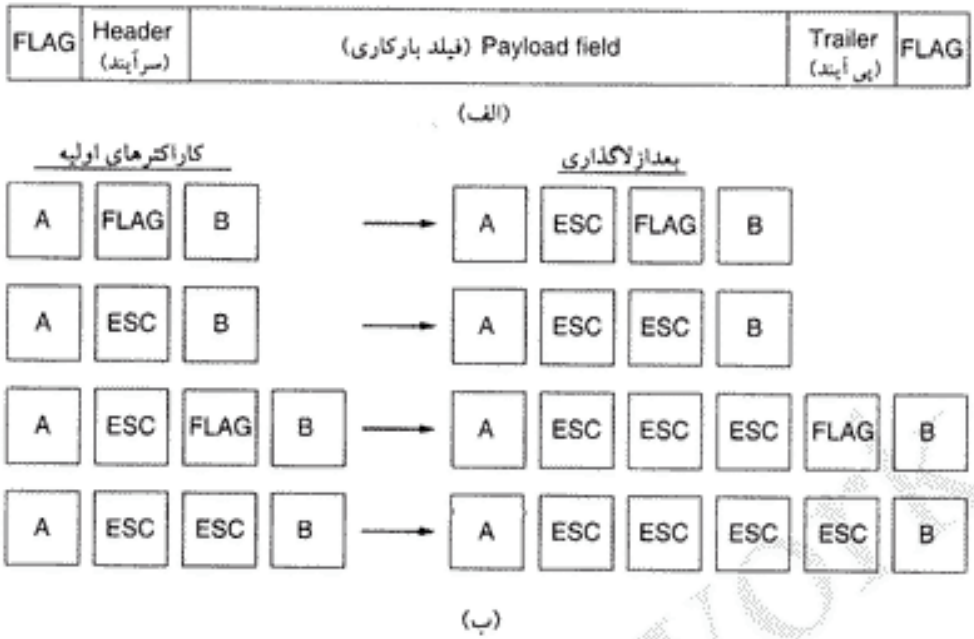
در اولین روش فریم‌بندی تعداد کاراکترهای فریم در یکی از فیلدهای سرآیند آن نوشته می‌شود. وقتی این فریم به مقصد می‌رسد، لایهٔ پیوند داده می‌تواند به کمک این فیلد ابتدا و انتهای فریم را مشخص کند. در شکل ۳-۴(الف) چهار فریم با تعداد کاراکترهای ۵، ۵، ۸ و ۸ را می‌بینید.

اشکال این روش آنست که فیلد تعداد کاراکترها نیز می‌تواند دچار خطا شود. برای مثال در شکل ۳-۴(ب)، فیلد تعداد کاراکترها در فریم دوم از ۵ به ۷ تبدیل شده است، و ماشین مقصد دیگر قادر نیست فریمهای بعدی را بدرستی بخواند (چون قادر نیست ابتدای آنها را تشخیص دهد). حتی اگر جمع تطبیقی اشتباه باشد و ماشین مقصد متوجه باشد که خطایی رخ داده، باز هم تشخیص نقطهٔ شروع بعدی برای آن غیرممکن است. درخواست ارسال مجدد نیز کمکی نمی‌کند، چون ماشین مقصد نمی‌داند فریمها تا کجا درست بوده، و اشتباه از کجا رخ داده است و نیاز به ارسال مجدد دارد.

مشکل سنکرون شدن مجدد مبدأ و مقصد بعد از بروز خطا در روش دوم فریم‌بندی (بایت‌های پرچم، با لاگذاری بایت) حل شده است، بدین ترتیب که هر فریم با تعدادی بایت خاص شروع و پایان می‌یابد. در گذشته، بایتهای شروع و پایان متفاوت بودند، ولی در سالهای اخیر از بایتهای یکسانی بعنوان بایت پرچم (flag byte) در



شکل ۳-۴. استریم کاراکترها. (الف) بدون خطا. (ب) با خطا.



شکل ۳-۵. (الف) تعیین ابتدا و انتهای فریم با استفاده از بایت پرچم. (ب) چهار توالی بایت قبل و بعد از لاگذاری بایت.

شروع و پایان فریم استفاده می‌شود. این بایتها را در شکل ۳-۵ (الف) با عنوان FLAG ملاحظه می‌کنید. در این روش اگر گیرنده همزمانی خود با فرستنده را از دست بدهد، فقط کافیست با جستجوی بایت پرچم انتهای فریم فعلی را پیدا کند. دو بایت پرچم که پشت سر هم بیایند، بمعنای پایان یک فریم و شروع فریم بعدی هستند.

یکی از مشکلات جدی این روش آنست که طرح بیت بایت پرچم می‌تواند در داده‌های اصلی نیز وجود داشته‌باشد (بویژه اگر اطلاعات از نوع برنامه‌های اجرایی یا اعداد اعشاری باشد). این وضعیت گیرنده را به اشتباه خواهد انداخت. یکی از راه‌های حل این وضعیت آنست که پروتکل لایهٔ پیوند داده در سمت فرستنده قبل از هر توالی بیت پرچم که در دادهٔ اصلی ظاهر می‌شود، یک **بایت گریز** (escape byte) خاص قرار دهد. لایهٔ پیوند داده مقصد این بایت‌ها را حذف کرده، و داده‌های اصلی را به لایهٔ شبکه تحویل می‌دهد. به این تکنیک **لاگذاری بایت** (byte stuffing) یا **لاگذاری کاراکتر** (character stuffing) گفته می‌شود. با این روش بایت پرچم بآسانی قابل تشخیص است، چون قبل از آن بایت گریز وجود ندارد.

اما حالا سؤال دیگری پیش می‌آید: اگر در وسط دادهٔ اصلی طرحی مشابه بایت گریز وجود داشت، چه اتفاقی می‌افتد؟ جواب اینست که قبل از این بایت هم یک بایت گریز قرار داده می‌شود؛ بعبارت دیگر دو بایت گریز پشت سر هم یعنی یک بایت گریز در دادهٔ اصلی. در شکل ۳-۵(ب) چند نمونه از حالتهایی که می‌تواند پیش آید، آورده شده است. در هر مورد آن چیزی که گیرنده می‌گیرد، دقیقاً مشابه آن چیزیست که فرستنده ارسال کرده است.

تکنیک لاگذاری بایت که در شکل ۳-۵ نشان داده شده، شکل ساده شدهٔ آن چیزیست که در پروتکل PPP (یکی از مهمترین پروتکلهای ارتباط با اینترنت در کامپیوترهای شخصی) مورد استفاده قرار می‌گیرد. (بعداً در همین فصل دربارهٔ PPP صحبت خواهیم کرد.)

یکی از معایب بزرگ فریم‌بندی بایت‌های پرچم با لاگذاری بایت وابستگی شدید آن به کاراکترهای ۸-بیتی است، و همانطور که می‌دانید تمام کُدها ۸-بیتی نیستند (برای مثال، در استاندارد **یونی‌کُد** از کاراکترهای ۱۶-بیتی استفاده می‌شود). فرض ۸-بیتی بودن کاراکترها در مکانیزم فریم‌بندی یکی از مشکلات جدی آن محسوب می‌شود، بهمین دلیل روش جدیدی که در آن طول کاراکتر می‌تواند متغیر باشد، ابداع شده است.

در این روش جدید طول کاراکترها اهمیتی ندارد، و فریمها می‌توانند تعداد بیتهای دلخواه داشته باشند. طرز کار این تکنیک جدید چنین است: هر فریم با طرح بیت خاصی (01111110 - که در واقع یک بایت پرچم است) شروع می‌شود. هرگاه لایهٔ پیوند داده در سمت فرستنده پنج 1 پشت سر هم در دادهٔ اصلی دید، بطور خودکار یک 0 بعد از آن قرار می‌دهد. این روش، که به آن **لاگذاری بیت** (bit stuffing) گفته می‌شود، بسیار شبیه لاگذاری بایت است.

وقتی گیرنده پنج 1 متوالی ببیند که یک 0 پشت سر آنها آمده، بطور خودکار این 0 را حذف می‌کند. لاگذاری بیت نیز مانند لاگذاری بایت بکلی از دید لایهٔ شبکه در هر دو کامپیوتر پنهان (شفاف) است. اگر در دادهٔ کاربر طرح بیت 01111110 وجود داشته باشد، لایهٔ پیوند داده فرستنده آنرا به 011111010 تبدیل می‌کند، و در سمت گیرنده این 0 اضافی حذف شده و طرح بیت 01111110 به لایهٔ بالاتر تحویل داده می‌شود. به یک مثال در شکل ۳-۶ توجه کنید.

(الف) 0110111111111111111110010

(ب) 0110111110111110111110010010

بیت های لاگذاری

(ج) 0110111111111111111110010

شکل ۳-۶. لاگذاری بیت. (الف) دادهٔ اولیه. (ب) داده‌ها بصورتی که روی خط فیزیکی ارسال می‌شود. (ج) داده‌ها بصورتی که در گیرنده دریافت می‌شود.

در روش لاگذاری بیت نیز محدودهٔ فریم با استفاده از پرچمهای شروع و پایان مشخص می‌شود، و گیرنده می‌تواند از آنها برای سنکرون شدن با فرستنده استفاده کند.

آخرین روش فریم‌بندی فقط در شبکه‌هایی قابل بکارگیری است که در کُدگذاری لایهٔ فیزیکی آنها نوعی **افزونگی** (redundancy) وجود داشته باشد. برای مثال در برخی از شبکه‌های LAN هر بیت داده با دو بیت فیزیکی نمایش داده می‌شود: بیت 1 با زوج بالا-پائین، و بیت 0 با زوج پائین-بالا. بدین‌ترتیب هر بیت داده دارای نوعی تغییر ولتاژ است، که تشخیص آنرا برای گیرنده ساده‌تر می‌کند. در چنین شبکه‌هایی زوج بالا-بالا و پائین-پائین برای داده‌ها استفاده نمی‌شود، و می‌توان از آنها برای مشخص کردن محدودهٔ فریمها سود برد.

لازم به ذکر است که در بسیاری از پروتکل‌های لینک داده برای اطمینان بیشتر از ترکیب روش شمارش کاراکترها با یکی دیگر از تکنیکهای گفته شده استفاده می‌شود. در این روش، انتهای فریم با استفاده از فیلد تعداد کاراکترها مشخص می‌شود، ولی فقط زمانی مورد قبول قرار می‌گیرد که جمع تطبیقی فریم نیز معتبر بوده و در این نقطه طرح بیت پایان فریم وجود داشته باشد. اگر چنین نباشد، گیرنده طرح بیت پایان فریم را در نقاط دیگر جستجو خواهد کرد.

۳-۱-۳ کنترل خطا

بعد از حل مسئلهٔ ابتدا و انتهای فریمها، نوبت به مسئلهٔ بعدی می‌رسد: چگونه می‌توان تمام فریمها را سالم و با ترتیب صحیح به مقصد رساند؟ فرض کنید فرستنده فقط فریمها را می‌فرستد و کاری ندارد که آنها به مقصد می‌رسند یا خیر. این وضعیت برای سرویسهای غیرمتصل بدون تصدیق دریافت خوب است، ولی برای سرویسهای قابل اعتماد (مانند سرویس اتصال-گرا با تصدیق دریافت) مسلماً خوب نیست.

یک سرویس قابل اعتماد باید بنحوی از رسیدن بسته‌ها به مقصد و آنچه در آنجا اتفاق می‌افتد، مطلع شود. معمولاً در این موارد پروتکل درخواست می‌کند که یک فریم کنترلی خاص (که محتوی تصدیق یا عدم تصدیق

دریافت صحیح فریمهاست) به فرستنده باز پس فرستاده شود. اگر فرستنده تصدیق مثبت دریافت کند، مطمئن می‌شود که فریم به سلامت به مقصد رسیده است. اما تصدیق منفی نشان می‌دهد که اوضاع روبراه نیست، و فریم باید مجدداً فرستاده شود.

مشکل دیگر اینجاست که گاهی (در اثر اشکالات سخت‌افزاری) یک فریم بکلی گم و ناپدید می‌شود. در این حالت گیرنده هیچ عکس‌العملی نشان نمی‌دهد، چون اساساً چیزی نگرفته که عکس‌العمل نشان دهد. بروشنی پیداست که در این حالت پروتکل سمت فرستنده تا ابد منتظر دریافت تصدیق از گیرنده می‌شود، تصدیقی که هرگز نخواهد رسید.

این مشکل را می‌توان با تعبیهٔ یک تایمر در لایهٔ پیوند داده حل کرد. وقتی فرستنده فریمی را می‌فرستد، تایمر را هم راه‌اندازی می‌کند. زمانی که این تایمر اندازه می‌گیرد آنقدر طولانی هست که بتوان با اطمینان گفت «فریم باید به مقصد رسیده، و تصدیق دریافت آن برگشته باشد». اگر همه چیز خوب پیش رفته باشد، معمولاً قبل از اینکه زمان تایمر به انتها برسد، تصدیق دریافت فریم به فرستنده برمی‌گردد (و تایمر ریست می‌شود).

اگر فریم یا پاسخ آن در راه گم شوند، تایمر در انتهای زمان مقرر اخطار می‌دهد؛ و ساده‌ترین راه حل همانا ارسال مجدد فریم است. اما ارسال چندبارهٔ فریمها این خطر را در بر دارد که چند تا از این فریمهای یکسان به مقصد برسند و به لایهٔ شبکه تحویل داده شوند. برای اجتناب از این وضعیت، فرستنده به هر فریم یک شمارهٔ ترتیبی می‌دهد تا گیرنده بتواند فریمهای مشابه و تکراری را تشخیص دهد.

با تمام این تمهیدات (تایمر و شمارهٔ ترتیبی فریمها) می‌توان مطمئن بود که از هر فریم یک (و فقط یک) نسخه به لایهٔ شبکه می‌رسد - و این یکی از مهمترین وظایف لایهٔ پیوند داده است. در ادامهٔ این فصل خواهید دید که لایهٔ پیوند داده چگونه این وظیفه را انجام می‌دهد.

۴-۱-۳ کنترل جریان

یکی دیگر از مسائل مهم در طراحی لایهٔ پیوند داده (و لایه‌های بالاتر) اینست که با فرستنده‌هایی که سریعتر از توان دریافت گیرنده مبادرت به ارسال اطلاعات می‌کنند، چه باید کرد؟ اگر کامپیوتر طرف فرستنده قویتر از گیرنده باشد (و یا بار کاری کمتری داشته باشد)، این وضعیت براحتی می‌تواند پیش بیاید. در این حالت گیرنده در سیلاب فریمهای ارسالی از فرستنده غرق می‌شود. حتی اگر کانال ارتباطی کاملاً عاری از خطا باشد، لحظه‌ای می‌رسد که گیرنده دیگر قادر به پردازش فریمهای ارسال شده نیست، و برخی از آنها را از دست می‌دهد. روشن است که باید کاری برای جلوگیری از این وضعیت کرد.

دو رهیافت برای مقابله با این وضعیت بکار گرفته می‌شود. در رهیافت اول، که **کنترل جریان بر اساس بازخور** (feedback-based flow control) نام دارد، این گیرنده است که آمادگی خود را برای دریافت اطلاعات بیشتر به فرستنده اعلام می‌کند (یا حداقل اعلام می‌کند در چه وضعیتی است). در رهیافت دوم، **کنترل جریان بر اساس نرخ** (rate-based flow control)، پروتکل مکانیزمی دارد که بدون استفاده از بازخور گیرنده نرخ ارسال اطلاعات را محدود می‌کند. در این فصل با روشهای کنترل جریان بر اساس بازخور آشنا می‌شوید، ولی از آنجائیکه رهیافت دوم هرگز در لایهٔ پیوند داده کاربرد ندارد، توضیح دربارهٔ آنرا به فصل ۵ موکول می‌کنیم.

انواع مختلفی از کنترل جریان بر اساس بازخور وجود دارد، ولی همهٔ آنها اصول مشترکی دارند: پروتکل قواعد تعریف شده‌ای دارد که زمان ارسال فریم بعدی را مشخص می‌کنند. طبق این قواعد فرستنده نمی‌تواند فریم بعدی را بفرستد، مگر اینکه (بطور صریح یا ضمنی) اجازهٔ گیرنده را دریافت کرده باشد. مثلاً، وقتی اتصال برقرار می‌شود، گیرنده می‌تواند به فرستنده بگوید: «اکنون می‌توانی n فریم بفرستی، ولی بعد از آن تا اجازه نداده‌ام چیزی نفرست.»

۲-۳ کشف و تصحیح خطا

همانطور که در فصل ۲ دیدید، سیستم تلفن سه بخش عمده دارد: سوئیچها، ترانکها، و حلقه‌های محلی. در اکثر کشورهای توسعه یافته دو بخش اول تماماً دیجیتال هستند. اما قسمت اعظم حلقه‌های محلی کماکان آنالوگ است، که بایستی با صرف هزینه‌های هنگفت در آینده به دیجیتال تبدیل شود. با اینکه در بخش دیجیتال خطا بندرت روی می‌دهد، نرخ آن در حلقه‌های محلی آنالوگ همچنان بالاست. علاوه بر آن، مخابرات بیسیم نیز بسرعت گسترش می‌یابد، که نرخ خطا در این قبیل سیستمها چندین برابر کانالهای فیبر نوری است. نتیجهٔ اخلاقی: فعلاً تا مدتها باید با خطاهای انتقال در سیستمهای مخابراتی بسازیم. در این قسمت خواهید دید چگونه.

خصلت خطا به منبع آن بستگی دارد؛ برای مثال، در سیستمهای رادیویی خطا بصورت **فورانی** (burst) رخ می‌دهد، نه تکی. این نوع خطا مزایا و معایبی دارد. توجه داشته باشید که کامپیوترها اطلاعات خود را بصورت بسته‌ای ارسال می‌کنند. اگر هر بستهٔ داده ۱۰۰۰ بیت و نرخ خطا نیز ۱ در ۱۰۰۰ باشد، می‌توان انتظار داشت که (در حالت غیرفورانی) تقریباً تمام بسته‌ها با خطا به مقصد برسند. اما اگر خطا بصورت فورانهای ۱۰۰ بیتی رخ دهد، بطور متوسط فقط یک یا دو بسته را خراب خواهد کرد. عیب بزرگ خطاهای فورانی آنست که کشف و تصحیح خطا در آنها بسیار دشوارتر است.

۱-۲-۳ کُدهای تصحیح خطا

طراحان شبکه دو استراتژی کلی برای مقابله با خطاها توسعه داده‌اند. یک راه اضافه کردن اطلاعات پراکنده به هر بلوک از داده‌هاست، بطوریکه گیرنده بتواند دادهٔ واقعی را از آن استنتاج کند. در روش دیگر فقط آنقدر اطلاعات اضافی به دادهٔ اصلی اضافه می‌شود که گیرنده از وقوع یا عدم وقوع خطا آگاهی یابد، و در صورت لزوم تکرار ارسال را خواستار شود. استراتژی اول **کُدهای تصحیح خطا** (error-correcting codes) و استراتژی دوم **کُدهای کشف خطا** (error-detecting codes) نام دارند. به کاربرد کُدهای تصحیح خطا اغلب **تصحیح پیشگیرانهٔ خطا** نیز گفته می‌شود.

هر یک از این تکنیکها جایگاه خاص خود را دارند. در کانالهای قابل اطمینان، مانند فیبر نوری، مقرون‌بصرفه‌تر است که از کُدهای کشف خطا استفاده کرده و بسته‌های معدودی را که خراب می‌شوند، دوباره ارسال کنیم. اما در کانالهایی مانند لینکهای بیسیم که پر از خطا هستند، بهتر است از تکنیکهای تصحیح خطا استفاده کرده و اجازه دهیم گیرنده خود دادهٔ واقعی را بدست آورد (چون باحتمال زیاد ارسال مجدد بسته‌ها هم عاری از خطا نخواهد بود).

برای مقابله با خطاها، ابتدا باید بدانیم خطا واقعاً چیست. معمولاً، یک فریم m بیت دادهٔ اصلی (یعنی، پیام) و r بیت دادهٔ پراکنده (یا اطلاعات چک‌کننده) دارد، که در مجموع n بیت می‌شود ($n = m + r$). به این واحد nبیتی (داده‌های اصلی و پراکنده) اغلب **کلمهٔ کُد** nبیتی گفته می‌شود.

دو کلمهٔ کُد 10001001 و 10110001 را در نظر بگیرید: براحتی می‌توان مشخص کرد که این دو کلمه چند اختلاف دارند. در این مورد ۳ بیت اختلاف وجود دارد. برای تعیین تعداد اختلاف‌ها می‌توان دو کلمهٔ کُد را با هم XOR (OR انحصاری) کرد، و تعداد 1 ها را شمرد:

```
10001001
10110001
--------
00111000
```

به تعداد اختلافهای دو کلمهٔ کُد **فاصلهٔ همینگ** (Hamming distance) گفته می‌شود (Hamming, 1950). اهمیت این فاصله در آنجاست که می‌توان ثابت کرد برای تبدیل شدن اتفاقی دو کلمه با فاصلهٔ d، بایستی d خطای تک‌بیتی روی دهد.

در اکثر سیستمهای انتقال، تمامی 2^m حالت ممکنهٔ دادهٔ اصلی مجاز است، ولی بدلیل روش محاسبهٔ بیت‌های افزونگی، تمام 2^n حالت کلمهٔ کُد مجاز نیست. با توجه به الگوریتم محاسبهٔ بیت‌های افزونگی، می‌توان لیستی از تمام حالتهای مجاز کلمهٔ کُد بدست آورد، و از این لیست دو کلمه‌ای که کمترین فاصلهٔ همینگ را دارند، پیدا کرد. این فاصله فاصلهٔ همینگ الگوریتم یا کُد موردنظر است.

خصوصیات تصحیح خطا یا کشف خطای یک کُد به فاصلهٔ همینگ آن بستگی دارد. برای کشف d خطا، به کُدی با فاصلهٔ همینگ $d + 1$ نیاز داریم، چون با چنین کُدی هیچ d خطای تک‌بیتی وجود ندارد که بتواند یک کلمهٔ کُد مجاز را به کلمهٔ کُد مجاز دیگر تبدیل کند. اگر گیرنده کلمهٔ کُد غیرمجازی دریافت کرد، می‌تواند با اطمینان بگوید که خطایی رخ داده است. بهمین‌ترتیب، برای تصحیح d خطا، به کُدی با فاصلهٔ $2d + 1$ نیاز داریم، چون در این حالت کلمات کُد چنان از هم فاصله دارند که حتی با بروز d خطا، کلمهٔ کُد خراب شده هنوز نزدیکترین فاصله را با کلمهٔ کُد اصلی دارد، و تشخیص آن براحتی ممکن است.

بعنوان نمونه‌ای از کُدهای کشف خطا، کُدی با یک **بیت توازن** (parity bit) را در نظر بگیرید. این بیت توازن بگونه‌ای انتخاب می‌شود که تعداد بیت‌های 1 کلمهٔ کُد همواره زوج (یا فرد) شود. برای مثال، اگر بخواهیم کلمهٔ 1011010 را با **توازن زوج** (even parity) ارسال کنیم، یک بیت 0 به انتهای آن اضافه می‌کنیم (10110100) ؛ اما اگر بخواهیم همین کلمه را با **توازن فرد** (odd parity) ارسال کنیم، باید یک بیت 1 به انتهای آن اضافه کنیم (10110101) . کُدی با یک بیت توازن دارای فاصلهٔ همینگ 2 است، چون هر خطای تک‌بیتی کلمهٔ کُدی با توازن اشتباه تولید می‌کند. این کُد می‌توان یک خطا در هر کلمه را آشکار کند.

بعنوان یک نمونهٔ ساده از کُدهای تصحیح خطا، کُدی را در نظر بگیرید که فقط چهار کلمهٔ کُد مجاز دارد:

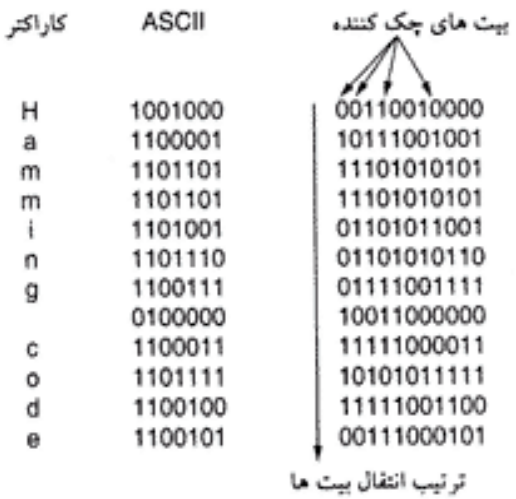
0000000000, 0000011111, 1111100000, 1111111111

فاصلهٔ همینگ این کُد 5 است، بنابراین می‌تواند دو خطا را تصحیح کند. اگر گیرنده کلمهٔ کُدی بصورت 0000000111 دریافت کند، می‌داند که کلمهٔ اصلی باید 0000011111 بوده باشد. اما اگر سه خطا کلمهٔ 0000000000 را به 0000000111 تبدیل کرده باشد، دیگر نمی‌توان آنرا بدرستی تصحیح کرد.

فرض کنید می‌خواهیم کُدی با m بیت دادهٔ اصلی و r بیت افزونگی طراحی کنیم که بتواند تمام خطاهای تک‌بیتی را تصحیح کند. هر یک از 2^m پیام مجاز دارای n کلمهٔ کُد غیرمجاز است که با آن 1 فاصله دارد (این را می‌توان بسادگی از معکوس کردن هر یک از بیت‌های کلمهٔ کُد nبیتی فهمید). بنابراین هر یک از 2^m پیام مجاز به طرحی اختصاصی با $n + 1$ بیت نیاز دارد. از آنجائیکه تعداد ترکیبات ممکنهٔ کلمهٔ کُد 2^n است، بایستی داشته باشیم: $(n + 1)2^m \leq 2^n$. با قرار دادن $n = m + r$ در این رابطه، داریم: $(m + r + 1) \leq 2^n$. با داشتن m، از این رابطه حداقل بیت‌های افزونگی لازم (r) برای تصحیح خطاهای تک‌بیتی بدست می‌آید.

همینگ در یکی از مقالات خود (1950) روشی برای بدست آوردن این حداقل معرفی کرد. وی بیت‌های کلمهٔ کُد را از چپ براست شماره گذاری کرد. بیت‌هایی که توانهایی از 2 هستند (1 ، 2 ، 4 ، 8 ، 16 ، و غیره)، بیت‌های چک‌کننده‌اند؛ سایر بیت‌ها (3 ، 5 ، 6 ، 7 ، 9 ، و غیره) بیت‌های پیام (m) هستند. هر بیت چک‌کننده توازن مجموعه‌ای از بیت (از جمله خودش) را زوج (یا فرد) می‌کند. هر بیت می‌تواند در بیش از یک مجموعهٔ توازن محاسبه شود. برای دیدن اینکه کدام بیت‌های چک‌کننده در محاسبهٔ توازن بیت داده‌ای در موقعیت k دخالت دارند، k را بصورت مجموع توانهای 2 می‌نویسیم. برای مثال، $11 = 1 + 2 + 8$ ، و $29 = 1 + 4 + 8 + 16$. هر بیت فقط با بیتهای چک‌کننده‌ای که در موقعیت‌های بدست آمده از مجموع توانهای 2 قرار دارند، چک می‌شود (مثلاً، بیت موقعیت 11 فقط با بیت‌های چک‌کنندهٔ 1 ، 2 و 8 چک می‌شود).

وقتی یک کلمهٔ کُد به گیرنده می‌رسد، گیرنده یک شمارنده را 0 می‌کند. سپس تمام بیت‌های چک‌کننده (k) را



شکل ۳-۷. استفاده از کُد همینگ برای تصحیح خطاهای فورانی.

از نظر توازن چک می‌کند ($k = 1, 2, 4, 8,...$). اگر توازن k درست نباشد، گیرنده k را به شمارنده اضافه می‌کند. اگر پس از پایان این عملیات شمارنده همچنان 0 باشد، کلمهٔ کُد صحیح تلقی و قبول می‌شود. اگر شمارنده 0 نباشد، حتماً شمارهٔ بیت خطا را نشان می‌دهد. برای مثال، اگر توازن بیت‌های چک‌کنندهٔ 1، 2 و 8 اشتباه باشد، بیت 11 غلط است، چون این تنها بیتی است که با بیت‌های چک‌کنندهٔ 1، 2 و 8 چک می‌شود. در شکل ۳-۷ چند کاراکتر آسکی ۷-بیتی را که با کُد همینگ ۱۱-بیتی کُد شده‌اند، می‌بینید. فراموش نکنید که داده‌های اصلی در موقعیتهای 3، 5، 6، 7، 9، 10 و 11 قرار دارند.

کُدهای همینگ فقط می‌توانند خطاهای تک‌بیتی را تصحیح کنند. با این حال روشی وجود دارد که اجازه می‌دهد تا این کُد خطاهای فورانی را نیز تصحیح کند. در این روش k کلمهٔ کُد متوالی بصورت ماتریس (یک کلمهٔ کُد در هر سطر) چیده می‌شوند. معمولاً، این کلمات تک به تک (از چپ براست) ارسال می‌شوند. برای تصحیح خطاهای فورانی، بایستی داده‌ها را بصورت ستونی (باز هم از چپ براست) ارسال کرد. وقتی k بیت اول (ستون اول) ارسال شد، نوبت به ستون دوم (و سپس ستونهای بعدی) می‌رسد (شکل ۳-۷ را ببینید). وقتی این فریم به گیرنده رسید، ماتریس از نو (ستون به ستون) ساخته می‌شود. اگر یک خطای فورانی به طول k رخ داده باشد، حداکثر یک بیت در هر کلمهٔ کُد تغییر خواهد کرد، و از آنجائیکه کُد همینگ می‌تواند یک خطا را تصحیح کند، تمام بلوک قابل تصحیح خواهد بود. در این روش برای مصون کردن km بیت داده در مقابل خطاهای فورانی با طول k (یا کمتر)، از kr بیت چک‌کننده استفاده شده است.

۳-۲-۲ کُدهای کشف خطا

کُدهای کشف خطا در لینکهای بیسیم، که در مقایسه با سیم مسی و فیبر نوری بطور وحشتناکی نویزی هستند، کاربرد گسترده‌ای دارد. بدون این کُدها شاید اساساً نتوان چیزی روی این لینکها رد و بدل کرد. اما در سیمهای مسی و فیبرهای نوری نرخ خطا بسیار کمتر است، و تشخیص خطا و ارسال مجدد بسته‌هایی که (ندرتاً) خراب می‌شوند، کاملاً کفایت می‌کند.

بعنوان مثال، کانالی را در نظر بگیرید که نرخ خطا در آن 1 در 10^6 و خطاها غیرفورانی هستند؛ اندازهٔ هر بلوک را هم 1000 بیت فرض می‌کنیم. برای داشتن ویژگی تصحیح خطا، هر بلوک ۱۰۰۰ بیتی به ۱۰ بیت چک‌کننده نیاز دارد، بعبارت دیگر برای ارسال 1 Mb داده باید 10 kb اطلاعات افزونگی (بیت‌های چک‌کننده) را نیز بهمراه آن

بفرستیم. اما برای کشف خطا فقط یک بیت توازن در هر بلوک کافیست. در این روش بار اضافی کشف خطا + ارسال مجدد یک بلوک خراب برای 1 Mb داده فقط 2001 بیت است، که در مقایسه با 10,000 بیت کُد همینگ بسیار کمتر است.

اگر در هر بلوک از یک بیت توازن برای کشف خطا استفاده کنیم و یک خطای فورانی رخ دهد، احتمال اینکه بتوانیم خطا را کشف کنیم فقط ۵۰٪ است، که بهیچوجه قابل قبول نیست. اما با تشکیل ماتریسی با n ستون و k سطر (که در بالا توضیح دادیم) اوضاع بنحو قابل توجهی بهتر خواهد شد. در این روش برای هر ستون یک بیت توازن محاسبه، و در آخرین سطر ماتریس نوشته می‌شود. هنگام ارسال نیز این ماتریس بصورت ستونی فرستاده می‌شود. گیرنده بعد از دریافت کل ماتریس، تمام بیت‌های توازن را چک می‌کند؛ و اگر هر یک از این بیت‌ها غلط باشد، ارسال مجدد ماتریس را درخواست می‌کند. این کار تا زمانی که ماتریس بطور کامل و بدون خطای توازن به دست گیرنده برسد، تکرار خواهد شد.

روش فوق می‌تواند خطاهای فورانی با طول حداکثر n بیت را آشکار کند، چون در این حالت فقط یک بیت در هر ستون تغییر خواهد کرد. اما اگر یک خطای فورانی با طول $n + 1$ رخ دهد بگونه‌ایکه فقط بیت اول و آخر را تغییر دهد (و سایر بیت‌ها تغییر نکنند)، نمی‌توان آنرا کشف کرد، زیرا بیت اول و آخر در یک سطر قرار می‌گیرند و توازن این سطر بدون تغییر خواهد ماند. (یک خطای فورانی الزاماً بمعنای معکوس شدن تمام بیت‌ها نیست: فقط می‌توان از معکوس شدن بیت اول و آخر مطمئن بود.) اگر طول خطای فورانی خیلی زیاد باشد یا تعدادی خطای فورانی کوتاه و پشت سر هم رخ دهد، احتمال اینکه یکی از ستونها تصادفاً صاحب توازن درست شود، ۵۰٪ است، بنابراین احتمال اینکه چنین بلوکی (به اشتباه) صحیح تلقی شود، 2^{-n} خواهد بود.

با اینکه روش فوق در مواردی کفایت می‌کند، اما در عمل از روش دیگری استفاده می‌شود: **کُد چندجمله‌ای** (polynominal code)، که به CRC (**چک افزونگی چرخه‌ای** – Cyclic Redundancy Check) نیز معروفست. در کُدهای چندجمله‌ای مبنا بر این است که هر رشته یک چندجمله‌ایست با ضرایب 0 و 1. با این فرض، یک فریم k-بیتی معادلست با عبارتی k جمله‌ای، با ضرایب x^{k-1} تا x^0. این چندجمله‌ای از درجهٔ $k - 1$ است. باارزشترین بیت (منتهی‌الیه سمت چپ) ضریب x^{k-1} است، بیت بعدی ضریب x^{k-2}، و الی آخر. برای مثال، رشتهٔ 110001 دارای ۶ بیت است بنابراین نشاندهندهٔ یک شش جمله‌ایست با ضرایب 1، 1، 0، 0، 0، و 1، که می‌توان آنرا چنین نوشت: $x^5 + x^4 + x^0$.

محاسبات چندجمله‌ایها در مدول 2 (و طبق قوانین جبر میدان) انجام می‌شود. در جمع و تفریق ۲ بر ۱ نادیده گرفته می‌شود، بعبارت دیگر شبیه XOR است. برای مثال،

	10011011		00110011		11110000		01010101
+	11001010	+	11001101	-	10100110	-	10101111
	01010001		11111110		01010110		11111010

تقسیم درست مانند تقسیم باینری است، با این تفاوت که تفریق‌ها در مدول 2 (مانند بالا) انجام می‌شود.

در هنگام استفاده از روش کُد چندجمله‌ای، فرستنده و گیرنده بایستی از قبل بر سر یک **چندجمله‌ای مولد** (generator polynominal)، که آنرا $G(x)$ می‌نامیم، توافق کنند. باارزشترین (چپ‌ترین) و کم‌ارزشترین (راست‌ترین) بیت‌های چندجمله‌ای مولد باید 1 باشد. برای محاسبهٔ **مجموع چک** (checksum) یک فریم m-بیتی (که چندجمله‌ای متناظر با آن $M(x)$ است)، این فریم باید طولانیتر از چندجمله‌ای مولد باشد. ایده آنست که یک مجموع چک به انتهای فریم اصلی چسبانده شود، بگونه‌ایکه فریم حاصله بر $G(x)$ قابل تقسیم باشد. اگر

تقسیم این فریم بر $G(x)$ در سمت گیرنده باقیمانده آورد، معلوم می‌شود که خطایی رخ داده است.

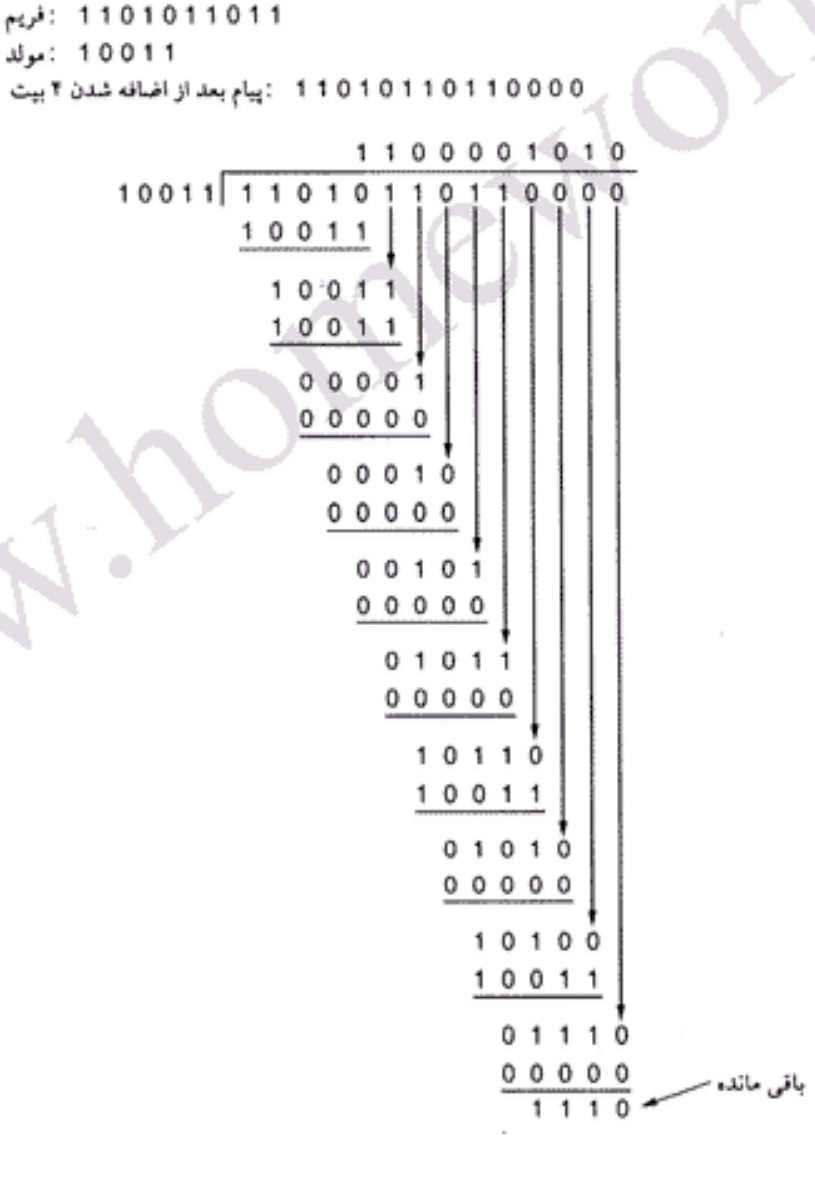
الگوریتم محاسبهٔ مجموع چک چنین است:

۱. فرض می‌کنیم چندجمله‌ای $G(x)$ از درجهٔ r است. r بیت 0 به سمت راست فریم اضافه می‌کنیم تا تعداد بیتهای آن به $m + r$ برسد. این چندجمله‌ای معادل $x^r M(x)$ خواهد شد.

۲. رشتهٔ $x^r M(x)$ را (در مدول 2) بر $G(x)$ تقسیم می‌کنیم.

۳. باقیمانده را (که همیشه r بیت یا کمتر دارد) از $x^r M(x)$ کم می‌کنیم (این تفریق هم در مدول 2 انجام می‌شود). حاصل تفریق همان فریم موردنظر (فریم اولیه + مجموع چک) است، که آنرا $T(x)$ می‌نامیم.

در شکل ۳-۸ طرز محاسبهٔ مجموع چک برای فریم 1101011011 را با مولد $G(x) = x^4 + x + 1$ ملاحظه می‌کنید.



شکل ۳-۸ محاسبهٔ مجموع چک کُد چندجمله‌ای.

همانطور که براحتی معلوم می‌شود، $T(x)$ بر $G(x)$ (در مدول 2) بخش‌پذیر است (چون وقتی باقیماندهٔ تقسیم را از مقسوم کم کنیم، عدد حاصله بطور حتم بر مقسوم‌علیه بخش‌پذیر خواهد بود). بطور مثال، اگر 210,278 را (در مبنای 10) بر 10,941 تقسیم کنیم، باقیمانده 2399 می‌شود که اگر آنرا از 210,278 کم کنیم، آنچه باقی می‌ماند (207,879)، بر 10,941 بخش‌پذیر خواهد بود.

حال اجازه دهید قدرت این روش را بررسی کنیم. این روش چه نوع خطاهایی را می‌تواند کشف کند؟ فرض کنید خطایی رخ داده، و بجای $T(x)$ رشتهٔ $T(x) + E(x)$ به گیرنده رسیده است، بطوریکه هر بیت 1 در $E(x)$ متناظر با یک بیت تغییر یافته است (بعبارت دیگر، اگر در این عبارت، $E(x)$، k بیت 1 وجود داشته باشد، k خطای تک‌بیتی رخ داده است). خطای فورانی نیز عبارتست از خطایی که با یک بیت 1 شروع و ختم شود، و بین آنها هر ترکیبی از 0 و 1 می‌تواند وجود داشته باشد.

وقتی این فریم به مقصد می‌رسد، گیرنده آنرا بر $G(x)$ تقسیم می‌کند (بعبارت دیگر $[T(x)+E(x)]/G(x)$ را محاسبه می‌کند). از آنجائیکه $T(x)/G(x) = 0$، این تقسیم معادل $E(x)/G(x)$ است. همانطور که می‌بینید، اگر خطای رخ داده دقیقاً طرحی شبیه $G(x)$ نداشته باشد، بطور مسلم آشکار خواهد شد.

فرض کنید یک خطای تک‌بیتی رخ داده است، یعنی، $E(x) = x^i$ (که i بیت خطاست). اگر $G(x)$ بیش از دو جمله داشته باشد، $E(x)$ هرگز بر آن بخش‌پذیر نخواهد بود - پس، این روش می‌تواند تمام خطاهای تک‌بیتی را آشکار کند.

اگر دو خطای تک‌بیتی جدا از هم رخ دهد، بطوریکه $E(x) = x^i + x^j$ (که در آن $i > j$)، می‌توان $E(x)$ را به صورت $x^j(x^{k-j} + 1)$ تجزیه کرد. اگر $G(x)$ بر x بخش‌پذیر نباشد، شرط کافی برای اینکه تمام خطاهای دوبیتی قابل کشف باشد آن است که $E(x)$ عبارت $x^k + 1$ را (برای تمام k های کوچکتر از $i - j$) بخش نکند. چندجمله‌ای‌های ساده و از درجهٔ پائینی می‌شناسیم که می‌توان با آنها فریمهای نسبتاً طویل را محافظت کرد. مثلاً، چندجمله‌ای $x^{15} + x^{14} + 1$ هیچ عبارت $x^k + 1$ را برای تمام k های کوچکتر از 32,768 بخش نمی‌کند.

اگر تعداد خطاهای رخ داده عددی فرد باشد، تعداد جملات $E(x)$ نیز فرد خواهد بود (برای مثال، تعداد جملات $x^5 + x^2 + 1$ فرد است، ولی $x^2 + 1$ چنین نیست). جالبست بدانید که هیچ چندجمله‌ای با تعداد جملات فرد وجود ندارد که (در مدول 2) بر $x + 1$ بخش‌پذیر باشد. بدین ترتیب اگر $G(x)$ را طوری انتخاب کنیم که بر $x + 1$ بخش‌پذیر باشد، می‌توانیم هر خطایی که تعداد بیتهای تغییر کرده فرد باشد را کشف کنیم.

برای اثبات اینکه هیچ چندجمله‌ای فرد وجود ندارد که بر $x + 1$ بخش‌پذیر باشد، فرض کنید $E(x)$ چندجمله‌ای فردیست که چنین خاصیتی دارد (بر $x + 1$ بخش‌پذیر است). اگر از $x + 1$ فاکتور بگیریم، $E(x)$ بصورت $(x + 1) Q(x)$ در می‌آید. حال $E(1) = (1 + 1) Q(1)$ را محاسبه می‌کنیم. از آنجائیکه (در مدول 2) $1 + 1 = 0$، $E(1)$ باید 0 باشد. اما اگر تعداد جملات $E(x)$ فرد باشد، قرار دادن 1 بجای x در آن همیشه نتیجهٔ 1 می‌دهد. بنابراین فرض ما نمی‌تواند درست باشد، و هیچ چندجمله‌ای فرد بر $x + 1$ بخش‌پذیر نیست.

بالاخره، و از همه مهمتر، یک کُد چندجمله‌ای با r بیت چک‌کننده تمام خطاهای فورانی با طول کمتر یا مساوی r را آشکار می‌کند. یک خطای فورانی با طول k را می‌توان با $x^i(x^{k-1} + \ldots + 1)$ نشان داد، که در آن i نقطهٔ شروع خطای فورانی از سمت راست فریم است. اگر مولد $G(x)$ دارای جملهٔ x^0 باشد، بر x^i بخش‌پذیر نخواهد بود؛ بنابراین اگر درجهٔ عبارت داخل پرانتز از درجهٔ $G(x)$ کمتر باشد، باقیماندهٔ تقسیم هرگز نمی‌تواند 0 شود.

اگر طول خطای فورانی $r + 1$ باشد، باقیماندهٔ تقسیم بر $G(x)$ صفر می‌شود فقط و فقط اگر طرح بیت خطا با $G(x)$ یکسان باشد. طبق تعریف بیت‌های اول و آخر خطای فورانی باید 1 باشند، بنابراین یکسان بودن آنها به $r - 1$ بیت میانی بستگی دارد. اگر تمام ترکیبات این $r - 1$ بیت را یکسان فرض کنیم، احتمال بروز این وضعیت $\frac{1}{2^{r-1}}$ خواهد بود.

همچنین می‌توان نشان داد که اگر طول خطای فورانی از $r + 1$ بزرگتر باشد یا چند خطای فورانی کوتاهتر رخ دهد، احتمال کشف نشدن خطا (با فرض یکسان بودن تمام ترکیبات ممکنه) $\frac{1}{2^r}$ است.

برخی از چندجمله‌ایها بصورت استاندارد بین‌المللی در آمده‌اند، که از میان آنها می‌توان به چندجمله‌ای زیر (که در IEEE 802 از آن استفاده می‌شود) اشاره کرد:

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$$

از ویژگیهای جالب این چندجمله‌ای آن است که هر نوع خطای فورانی با طول 32 یا کمتر، و خطاهای فورانی که تعداد بیتهای تغییر کرده فرد باشد، را آشکار می‌کند.

با اینکه بنظر می‌رسد محاسبهٔ مجموع چک و تست آن کار پیچیده‌ای باشد، پترسون و براون (1961) نشان دادند که می‌توان این کار را با یک مدار **شیفت رجیستر** (shift register) ساده بصورت سخت‌افزاری انجام داد. در واقع، این مدار در تمام کارتهای شبکه تعبیه شده است، و بسیاری از خطوط نقطه-به-نقطه هم از آن استفاده می‌کنند.

برای مدتهای مدید تصور بر آن بود که فریمهایی که مجموع چک آنها محاسبه می‌شود دارای طرح بیت تصادفی هستند، و تمام الگوریتمهای محاسبهٔ مجموع چک نیز فرض را بر این می‌گذاشتند. اما بررسی دقیق داده‌های واقعی نشان داده که این فرض بکلی اشتباه است. در نتیجه، خطاهایی که (تحت شرایط خاص) کشف نشده می‌مانند شایعتر از آن چیزیست که قبلاً تصور می‌شد (Partridge et al., 1995).

۳-۳ چند پروتکل سادهٔ لینک داده

برای آشنایی با پروتکل‌های لایهٔ پیوند داده، در این قسمت سه پروتکل را (که بتدریج پیچیده‌تر می‌شوند) بررسی خواهیم کرد. برای خوانندگان علاقمند، شبیه‌ساز این پروتکلها (و پروتکلهایی که در آینده خواهید دید) را در سایت وب کتاب قرار داده‌ایم (http://www.prenhall.com/tanenbaum). اما قبل از اینکه سراغ این پروتکلها برویم، اجازه دهید چند تا از فرض‌هایی را که دربارهٔ مدل ارتباطی زیربنایی داشته‌ایم، توضیح دهیم. اول اینکه فرض کرده‌ایم در لایه‌های فیزیکی، لایهٔ پیوند داده و شبکه پروسسهایی هستند که مستقل از یکدیگرند، و ارتباط آنها از طریق رد و بدل کردن پیام صورت می‌گیرد. در بسیاری از موارد، پروسسهای لایهٔ فیزیکی و لینک داده در پردازندهٔ کارت شبکه اجرا می‌شوند، و پروسسهای لایهٔ شبکه در CPU اصلی کامپیوتر. البته پیاده‌سازیهای دیگری نیز محتمل است (مثلاً، تمام پروسسهای لایهٔ فیزیکی، لینک داده و شبکه در پردازندهٔ کارت شبکه اجرا شوند، یا همگی آنها را CPU اصلی اجرا کند). در هر حال، مستقل دانستن این پروسسها بحث دربارهٔ آنها را بسیار ساده‌تر می‌کند، و تأکیدیست بر مستقل بودن لایه‌ها.

فرض کلیدی دیگر اینست که، ماشین A با استفاده از یک سرویس اتصال-گرای قابل اعتماد استریم طویلی از داده‌ها را به ماشین B می‌فرستد. بعدها این حالت که B هم همزمان به A داده بفرستد، را بررسی خواهیم کرد. علاوه بر آن، فرض کرده‌ایم ماشین A منبع بی‌پایانی از داده‌ها دارد که ارسال کند، و هرگز منتظر آمدن داده‌ها نخواهد شد. بعبارت دیگر، هر گاه لایهٔ پیوند داده A درخواست داده کند، لایهٔ شبکه بلافاصله اجابت می‌کند. (بعدها این فرض را هم کنار خواهیم گذاشت.)

فرض دیگر ما اینست که این کامپیوترها هرگز از کار نمی‌افتند؛ یعنی، پروتکلهای ما فقط با خطاهای مخابراتی سروکار دارند، نه هنگ کردن کامپیوتر و مسائلی از این قبیل.

دیگر اینکه، تا آنجا که به لایهٔ پیوند داده مربوط است، بسته‌ای که به لایهٔ شبکه داده می‌شود، **دادهٔ خالص** است، و باید تا آخرین بیت به آن تحویل شود. این که بخشی از این داده‌ها سرآیند بسته هستند و لایهٔ شبکه آنها را دور می‌ریزد، به خودش مربوط است نه به لایهٔ پیوند داده.

وقتی لایهٔ پیوند داده بسته‌ای از لایهٔ شبکه می‌گیرد تا ارسال کند، آنرا فریم‌بندی کرده و **سرآیند** (header) و **پی‌آیند** (trailer) های لازم را به آن می‌چسباند (شکل ۳-۱ را ببینید). بنابراین هر فریم از سه بخش تشکیل می‌شود: قسمتی از بسته‌ای که لایهٔ شبکه فرستاده، یک سرآیند شامل اطلاعات کنترلی، و یک پی‌آیند شامل مجموع چک فریم. سپس این فریم به لایهٔ پیوند داده ماشین مقصد فرستاده می‌شود. فرض ما بر این است که روال‌های کتابخانه‌ای *to_physical_layer* (برای ارسال) و *from_physical_layer* (برای دریافت) از قبل وجود دارند. مجموع چک نیز (مثلاً، با استفاده از کُدهای چندجمله‌ای) توسط سخت‌افزار محاسبه (و به انتهای فریم اضافه) می‌شود، بنابراین لازم نیست لایهٔ پیوند داده نگران آن باشد.

در ابتدا، لازم نیست گیرنده کاری انجام دهد؛ فقط منتظر می‌ماند تا اتفاقی بیفتد. در مثالهای این فصل، فرض کرده‌ایم که لایهٔ پیوند داده این کار را با روالی بنام *wait_for_event(&event)* انجام می‌دهد. این روال فقط وقتی به پایان می‌رسد (و کنترل را به برنامهٔ اصلی برمی‌گرداند) که اتفاقی افتاده باشد (یعنی، یک فریم دریافت شده باشد). اینکه چه اتفاقی افتاده است، را متغیر *event* مشخص می‌کند؛ و این که چه اتفاقهایی می‌تواند بیفتد، به تعریف پروتکل بستگی دارد. توجه داشته باشید که در دنیای واقعی لایهٔ پیوند داده (مانند این مثالها) در یک حلقهٔ بی‌انتها منتظر رسیدن فریمها نمی‌ماند، بلکه با استفاده از **وقفه** (interrupt) به آنها رسیدگی می‌کند. با این حال برای اجتناب از پیچیدگی مطلب، فرض کرده‌ایم که لایهٔ پیوند داده هیچ کار دیگری جز رسیدگی به کانال ارتباطی ما ندارد.

وقتی یک فریم به گیرنده می‌رسد، سخت‌افزار مجموع چک آنرا محاسبه می‌کند. اگر این مجموع چک اشتباه باشد (یعنی خطایی رخ داده)، به لایهٔ پیوند داده اطلاع داده می‌شود (*event = cksum_err*). اگر فریم بدرستی دریافت شده باشد، باز هم به اطلاع لایهٔ پیوند داده می‌رسد (*event = frame_arrival*). در این حالت لایهٔ پیوند داده با استفاده از تابع *from_physical_layer* فریم را گرفته، اطلاعات کنترلی موجود در سرآیند آنرا چک می‌کند، و اگر همه چیز مرتب باشد، سرآیند را جدا کرده و بخش اصلی داده را به لایهٔ شبکه تحویل می‌دهد.

تحت هیچ شرایطی سرآیند فریم به لایهٔ شبکه تحویل نمی‌شود، و برای این کار دلیل خوبی وجود دارد. پروتکلهای لینک داده و شبکه باید کاملاً از یکدیگر مستقل باشند. مستقل بودن پروتکلهای این دو لایه باعث می‌شود که بتوان هر کدام از این پروتکلها را تغییر داد، بدون اینکه نیاز باشد پروتکلهای لایهٔ دیگر تغییر کند (البته تحت هر شرایطی نحوهٔ تعامل و ارتباط لایه‌ها نباید تغییر کند). جدا و مستقل بودن لایه‌ها تا حد زیادی طراحی آنها را ساده می‌کند، چون می‌توان بدون نگرانی از اتفاقاتی که در لایه‌های دیگر می‌افتد، روی طراحی عملکردهای همان لایه تمرکز کرد.

در شکل ۳-۹ مقداری **تعریف** (به زبان C) می‌بینید، که در طراحی پروتکلهای این قسمت به آنها نیاز داریم. در اینجا پنج **ساختار داده** تعریف شده است: *boolean*، *seq_nr*، *packet*، *frame_kind* و *frame*. ساختار *boolean* از نوع **شمارشی** (enum) است، و می‌تواند دو مقدار *true* و *false* بگیرد. ساختار *seq_nr* از نوع **عدد صحیح بدون علامت** (unsigned int) تعریف شده، و برای شماره‌گذاری فریمها از آن استفاده خواهیم کرد. شماره‌گذاری فریمها از 0 تا *MAX_SEQ* (که بسته به نیاز هر پروتکل تعریف می‌شود) انجام می‌گیرد. *packet* (بسته) واحدی از اطلاعات است که بین لایهٔ شبکه و لایهٔ پیوند داده (روی یک ماشین، یا روی ماشین‌های جداگانه) رد و بدل می‌شود. در مدل ما هر بسته همیشه حاوی *MAX_PKT* بایت داده است، ولی به واقعیت نزدیکتر است که طول بسته را متغیر در نظر بگیریم.

هر *frame* از چهار فیلد تشکیل شده: *kind*، *seq*، *ack* و *info* - که سه تای اول اطلاعات کنترلی هستند، و آخری همان داده‌هائیست که باید منتقل شود. به مجموعهٔ فیلدهای کنترلی سرآیند فریم (frame header) گفته می‌شود.

```
#define MAX_PKT 1024    /* determines packet size in bytes */

typedef enum {false, true} boolean;    /* boolean type */
typedef unsigned int seq_nr;    /* sequence or ack numbers */
typedef struct {unsigned char data[MAX_PKT];} packet;    /*    packet definition */
typedef enum {data, ack, nak} frame_kind;    /* frame_kind definition */

typedef struct {    /* frames are transported in this layer */
  frame_kind kind;    /* what kind of a frame is it? */
  seq_nr seq;        /* sequence number */
  seq_nr ack;        /* acknowledgement number */
  packet info;       /* the network layer packet */
} frame;

/* Wait for an event to happen; return its type in event. */
void  wait_for_event(event_type *event);

/* Fetch a packet from the network layer for transmission on the channel. */
void  from_network_layer(packet *p);

/* Deliver information from an inbound frame to the network layer. */
void  to_network_layer(packet *p);

/* Go get an inbound frame from the physical layer and copy it to r. */
void  from_physical_layer(frame *r);

/* Pass the frame to the physical layer for transmission. */
void  to_physical_layer(frame *s);

/* Start the clock running and enable the timeout event. */
void  start_timer(seq_nr k);

/* Stop the clock and disable the timeout event. */
void  stop_timer(seq_nr k);

/* Start an auxiliary timer and enable the ack_timeout event. */
void  start_ack_timer(void);

/* Stop the auxiliary timer and disable the ack_timeout event. */
void  stop_ack_timer(void);

/* Allow the network layer to cause a  network_layer_ready event. */
```

```
void enable_network_layer(void);

/* Forbid the network layer from causing a network_layer_ready event. */
void disable_network_layer(void);

/* Macro inc is expanded in-line: Increment k circularly. */
#define inc(k) if (k < MAX_SEQ) k = k + 1; else k = 0
```

شکل ۳-۹. تعریف‌های موردنیاز برای پروتکلهایی که در این فصل می‌نویسیم. این تعریف‌ها در فایلی بنام *protocol.h* قرار داده می‌شوند.

فیلد *kind* می‌گوید که آیا داده‌ای در فریم وجود دارد یا خیر، چون برخی از پروتکلها فریمهای بدون داده را از فریمهایی که داده دارند، تمیز می‌دهند. فیلدهای *seq* و *ack* بترتیب برای شمارهٔ ترتیبی فریم و تصدیق دریافت مورد استفاده قرار می‌گیرند؛ بعداً در این باره بیشتر توضیح خواهیم داد. دادهٔ اصلی (بسته) در فیلد *info* فریم قرار دارد؛ فریمهای کنترلی نیز وجود دارند که اساساً در آنها فیلد *info* وجود ندارد. در پروتکلهای واقعی که طول فیلد *info* می‌تواند متغیر باشد، نیازی به تمایز بین فریمهای داده و فریمهای کنترلی نیست (چون فریم کنترلی فریمی است که طول فیلد *info* در آن 0 است).

در اینجا لازم است تفاوت فریم (frame) و بسته (packet) را مجدداً یادآور شویم. لایهٔ شبکه با گرفتن پیام از لایه انتقال و اضافه کردن سرآیند، آنرا بصورت بسته در می‌آورد. سپس این بسته به لایهٔ پیوند داده تحویل می‌شود، که در آنجا در فیلد *info* یک فریم قرار داده شده و برای ارسال آماده می‌شود. وقتی این فریم به لایهٔ پیوند داده ماشین مقصد رسید، بسته از فیلد *info* استخراج شده و به لایهٔ شبکه تحویل می‌شود. این فرآیند بکلی شفاف است، و لایه‌های شبکه در دو ماشین متقابل تصور می‌کنند که مستقیماً در حال تبادل بسته‌اند.

در برنامهٔ شکل ۳-۹ چند روال (تابع) نیز تعریف شده است. اینها روالهای کتابخانه‌ای هستند، که فقط کاری که انجام می‌دهند برای ما مهم است (و اصلاً اهمیتی ندارد این کار را چگونه انجام می‌دهند). روال *wait_for_event* (همانطور که قبلاً گفتیم) در یک حلقهٔ بی‌انتها به انتظار می‌ماند تا اتفاقی بیفتد. روالهای *to_network_layer* و *from_network_layer* بترتیب برای ارسال بسته به لایهٔ شبکه و برای گرفتن بسته از این لایه بکار می‌روند (اینها واسط لایه‌های ۲ و ۳ هستند). برای تبادل اطلاعات با لایهٔ فیزیکی نیز از روالهای *to_physical_layer* و *from_physical_layer* استفاده می‌شود (اینها واسط لایه‌های ۱ و ۲ هستند).

در پروتکلهای واقعی فرض بر اینست که کانال ارتباطی نامطمئن است، و احتمال این هست که فریمها در راه از بین بروند. برای مقابله با چنین وضعیتی، لایهٔ پیوند داده همزمان با ارسال هر فریم، باید یک تایمر را راه‌اندازی کند. اگر بعد از مدتی معین پاسخی از طرف مقابل نرسید، تایمر مزبور لایهٔ پیوند داده را (با استفاده از یک وقفه) مطلع می‌کند.

در پروتکلهای ما، در چنین وضعیتی روال *wait_for_event* مقدار *event = timeout* برمی‌گرداند. روالهای *start_timer* و *stop_timer* نیز بترتیب تایمر را روشن و خاموش می‌کنند (البته سپری شدن زمان مقتضی – *timeout* – فقط زمانی اتفاق می‌افتد که تایمر روشن باشد). یک تایمر را می‌توان (قبل از منقضی شدن آن) با اجرای مجدد روال *start_timer* ریست کرد.

روالهای *start_ack_timer* و *stop_ack_timer* تایمر دیگری را (برای ایجاد فریمهای تصدیق دریافت در شرایطی خاص) کنترل می‌کنند.

از روالهای *enable_network_layer* و *disable_network_layer* در پروتکلهای پیچیده‌تر استفاده می‌شود (ما در پروتکلهای سادهٔ این قسمت از این روالها استفاده نخواهیم کرد، چون فرض کرده‌ایم که لایهٔ شبکه همیشه می‌تواند به لایهٔ پیوند داده بسته تحویل دهد). وقتی لایهٔ پیوند داده لایهٔ شبکه را فعال می‌کند (*enable_network_layer*)، لایهٔ شبکه اجازه دارد آماده شدن بستهٔ داده را با یک وقفه به لایهٔ پیوند داده اطلاع دهد (این کار با *event = network_layer_ready* انجام می‌شود). اگر لایهٔ شبکه غیرفعال باشد (*disable_network_layer*)، اجازهٔ چنین کاری را ندارد. با استفاده دقیق و بجا از روالهای *enable_network_layer* و *disable_network_layer*، لایهٔ پیوند داده می‌تواند مطمئن شود که (هنگام پر شدن بافر) در سیلاب بسته‌های ارسالی از لایهٔ شبکه غرق نخواهد شد.

شمارهٔ ترتیبی فریم همیشه بین 0 تا *MAX_SEQ* (و از جمله خود این دو عدد) است، که البته *MAX_SEQ* در پروتکلهای مختلف می‌تواند متفاوت باشد. شمارهٔ ترتیبی فریمها معمولاً یکی یکی اضافه می‌شود، و وقتی به *MAX_SEQ* رسید، دوباره 0 خواهد شد - این کار بر عهدهٔ ماکروی *inc* گذاشته شده است. برای سرعت بخشیدن به اجرای این عملیات، *inc* بصورت **ماکرو** (macro) تعریف شده است. [کامپایلر با دیدن یک ماکرو، دستور معادل را جایگزین آن می‌کند، و مانند تابع آنرا فراخوانی نمی‌کند. م.] همانطور که بعداً خواهید دید، سرعت اجرای پروتکلها یکی از عوامل کلیدی در کارایی شبکه است، و استفاده از ماکرو (بجای تابع) تأثیر زیادی بر بهبود این کارایی دارد. همچنین، از آنجائیکه *MAX_SEQ* در پروتکلهای مختلف مقادیر متفاوتی دارد، تعریف *inc* بصورت ماکرو امکان می‌دهد تا بدون هیچ مشکلی از آن در پروتکلهای مختلف استفاده کنیم.

تعاریف شکل ۳-۹ بخشی از پروتکلهاییست که در قسمتهای آینده خواهیم نوشت. البته می‌توانستیم آنها را در ابتدای هر پروتکل نیز بیاوریم، ولی با جمع کردن آنها در یک فایل کُدهای آینده بسیار ساده‌تر خواهند شد. در زبان C، این کار با استفاده از دستور #include و نوشتن نام فایل تعاریف (در اینجا *protocol.h*) انجام می‌شود.

۳-۳-۱ پروتکل یکطرفهٔ نامقید

در اولین مثال یک پروتکل بسیار ساده را در نظر می‌گیریم، که در آن داده‌ها فقط در یک جهت منتقل می‌شوند. لایهٔ شبکه در فرستنده و گیرنده آمادهٔ کار هستند، زمان پردازش را می‌توان نادیده گرفت، از نظر بافر هیچ کمبودی وجود ندارد، و مهمتر از همه اینکه کانال ارتباطی بین دو لایهٔ پیوند داده کامل و بدون نقص است، و هیچ خطایی در آن رخ نمی‌دهد. این پروتکل غیرواقعی (که شاید «اتوپیا» - پروتکل آرمانی - مناسبترین نام برای آن باشد) را در شکل ۳-۱۰ ملاحظه می‌کنید.

```
/* Protocol 1 (utopia) provides for data transmission in one direction only, from
   sender to receiver. The communication channel is assumed to be error free
   and the receiver is assumed to be able to process all the input infinitely quickly.
   Consequently, the sender just sits in a loop pumping data out onto the line as
   fast as it can. */

typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender1(void)
{
```

```
  frame s;                        /* buffer for an outbound frame */
  packet buffer;                  /* buffer for an outbound packet */

  while (true) {
        from_network_layer(&buffer);     /* go get something to send */
        s.info = buffer;                 /* copy it into s for transmission */
        to_physical_layer(&s);           /* send it on its way */
  }                               /* Tomorrow, and tomorrow, and tomorrow,
                                     Creeps in this petty pace from day to day
                                     To the last syllable of recorded time.
                                        - Macbeth, V, v */
}

void receiver1(void)
{
  frame r;
  event_type event;               /* filled in by wait, but not used here */

  while (true) {
        wait_for_event(&event);          /* only possibility is frame_arrival */
        from_physical_layer(&r);         /* go get the inbound frame */
        to_network_layer(&r.info);       /* pass the data to the network layer */
  }
}
```

شکل ۳-۱۰. پروتکل یکطرفهٔ نامقید.

این پروتکل دارای دو روال مجزا است: فرستنده و گیرنده. فرستنده در لایهٔ پیوند داده ماشین مبدأ، و گیرنده در لایهٔ پیوند داده ماشین مقصد اجرا می‌شود. در اینجا از شمارهٔ ترتیبی فریمها و تصدیق دریافت استفاده‌ای نمی‌شود، بنابراین به *MAX_SEQ* هم نیازی نیست. تنها رویداد قابل انتظار *frame_arrival* (رسیدن صحیح و سالم فریم) است.

فرستنده یک حلقهٔ بی‌انتهای while است که داده‌ها را با حداکثر توان به بیرون پمپ می‌کند. بدنهٔ این حلقه سه کار انجام می‌دهد: آوردن یک بسته از لایهٔ شبکه (که همیشه آمادهٔ خدمت است)، ایجاد فریم خروجی با استفاده از متغیر *s*، و فرستادن فریم. این پروتکل فقط از فیلد *info* فریم استفاده می‌کند، چون فیلدهای دیگر مربوط به کنترل جریان و خطا هستند، که طبق فرض ما چنین محدودیتهایی در اینجا وجود ندارد.

گیرنده هم بهمان اندازه ساده است: انتظار بی‌پایان برای دریافت فریمی که همیشه سالم و بی‌نقص است. وقتی یک فریم از راه رسید، روال *wait_for_event* کنترل را به برنامهٔ اصلی برمی‌گرداند، و متغیر *event* را به *frame_arrival* ست می‌کند (که بهر حال استفاده‌ای از آن نمی‌شود). با فراخوانی روال *from_physical_layer*، فریم تازه از راه رسیده از بافر سخت‌افزاری برداشته شده و در متغیر *r* قرار داده می‌شود (تا گیرنده می‌تواند آنرا بردارد). در پایان، بخش دادهٔ این فریم (فیلد *info*) به لایهٔ شبکه فرستاده شده، و لایهٔ پیوند داده به انتظار فریم بعدی می‌نشیند.

۳-۳-۲ پروتکل توقف-انتظار یکطرفه

حال اجازه دهید غیرواقعی‌ترین بخش از پروتکل ۱ را کنار بگذاریم: نامحدود بودن توانایی این پروتکل در دریافت بسته‌ها از لایهٔ شبکه، و پردازش فریمهای ورودی (که به معنای نامحدود بودن بافر لایهٔ پیوند داده در سمت گیرنده است). اما کانال ارتباطی را همچنان بدون خطا، و ارتباط را یکطرفه فرض کرده‌ایم.

مهمترین مشکلی که با آن روبرو هستیم، این است که چگونه از غرق شدن گیرنده در سیلاب فریمهایی (که پردازش آنها از توان وی خارج است) جلوگیری کنیم. واضحست که، اگر گیرنده برای اجرای روالهای *from_physical_layer* و *to_network_layer* به زمان Δt نیاز داشته باشد، سرعت متوسط ارسال فرستنده بایستی از یک فریم بر Δt کمتر باشد. همچنین اگر فرض کنیم که سخت‌افزار گیرنده بطور خودکار عمل بافر کردن فریمها را انجام نمی‌دهد، فرستنده نباید قبل از برداشته شدن یک فریم از بافر لایهٔ فیزیکی (که توسط روال *from_physical_layer* انجام می‌شود)، فریم بعدی را ارسال کند چون در غیر اینصورت فریم قبلی از بین می‌رود (به این حالت **روهم‌نویسی** – overrun – می‌گویند).

در شرایط خاصی (مانند ارتباط سنکرون، و گیرنده‌ای که تنها وظیفهٔ آن گرفتن اطلاعات از خط ورودی است)، با ایجاد تأخیر در قسمت فرستندهٔ پروتکل ۱ و کُند کردن آن می‌توان به اهداف فوق دست یافت. اما بسیار محتملتر است که یک لایهٔ پیوند داده مجبور باشد چندین خط ورودی را پردازش کند، که در این حالت فاصلهٔ زمانی دریافت فریمها و پردازش آنها می‌تواند بسیار متغیر باشد. اگر طراحان شبکه بتوانند بدترین حالت گیرنده را محاسبه کنند، می‌توانند فرستنده را آنقدر کُند کنند که روهم‌نویسی هرگز اتفاق نیفتد. اما این روش بسیار محافظه کارانه است و بنحو بسیار بدی پهنای باند را تلف می‌کند، مگر اینکه تفاوت بهترین و بدترین حالت چندان زیاد نباشد (یعنی تفاوت پاسخهای لایهٔ پیوند داده ناچیز باشد).

راه حل بهتر این معضل، برگرداندن **بازخور** (feedback) از گیرنده به فرستنده است. بعد از تحویل بسته به لایهٔ شبکه، گیرنده یک فریم کوچک (که لازم نیست معنی خاصی هم داشته باشد) به فرستنده می‌فرستد، که در واقع مجوز ارسال فریم بعدی محسوب می‌شود. فرستنده بعد از ارسال یک فریم، آنقدر منتظر می‌ماند تا این فریم کوچک (که در واقع همان **تصدیق دریافت** – acknowledgement – است) از راه برسد. استفاده از بازخور گیرنده برای اطلاع به فرستنده (و دادن مجوز ارسال فریمهای بعدی) یکی از نمونه‌های **کنترل جریان** (flow control)، که قبلاً به آن اشاره کردیم، است.

پروتکلهایی که در آنها فرستنده قبل از ارسال فریم بعدی منتظر تصدیق دریافت فریم قبلی از گیرنده می‌ماند، به پروتکلهای **توقف-انتظار** (stop-and-wait) معروفند. در شکل ۳-۱۱ یک نمو از پروتکلهای توقف-انتظار را ملاحظه می‌کنید.

```
/* Protocol 2 (stop-and-wait) also provides for a one-directional flow of data from
   sender to receiver. The communication channel is once again assumed to be error
   free, as in protocol 1. However, this time, the receiver has only a finite buffer
   capacity and a finite processing speed, so the protocol must explicitly prevent
   the sender from flooding the receiver with data faster than it can be handled. */

typedef enum {frame_arrival} event_type;
"h.locotorp" edulcni#
```

```
void sender2(void)
{
  frame s;                          /* buffer for an outbound frame */
  packet buffer;                    /* buffer for an outbound packet */
  event_type event;                    /* frame_arrival is the only possibility */

  while (true) {
       from_network_layer(&buffer);     /* go get something to send */
       s.info = buffer;                 /* copy it into s for transmission */
       to_physical_layer(&s);           /* bye-bye little frame */
       wait_for_event(&event);          /* do not proceed until given the go ahead */
  }
}

void receiver2(void)
{
  frame r, s;                       /* buffers for frames */
  event_type event;                    /* frame_arrival is the only possibility */
  while (true) {
       wait_for_event(&event);          /* only possibility is frame_arrival */
       from_physical_layer(&r);            /* go get the inbound frame */
       to_network_layer(&r.info);       /* pass the data to the network layer */
       to_physical_layer(&s);           /* send a dummy frame to awaken sender */
  }
}
```

شکل ۳-۱۱. پروتکل توقف-انتظار یکطرفه.

با اینکه این پروتکل **یکطرفه** (simplex) است (یعنی ما فقط در یک جهت ارسال می‌کنیم)، اما فریمها می‌توانند در هر دو جهت رفت و آمد کنند. برای این منظور لازم است کانال ارتباطی ما چنین قابلیتی داشته باشد؛ البته یک کانال **دوطرفهٔ ناهمزمان** (half-duplex) هم کفایت می‌کند، چون فرستنده و گیرنده در آن واحد اقدام به فرستادن فریمها نمی‌کنند: فرستنده یک فریم می‌فرستد، گیرنده پاسخ می‌دهد، فرستنده فریم بعدی را می‌فرستد، گیرنده پاسخ می‌دهد، و الی آخر.

در اینجا هم (مانند پروتکل ۱) فرستنده همان سه کار قبلی را انجام می‌دهد: آوردن یک بسته از لایهٔ شبکه، ایجاد فریم خروجی، و فرستادن آن. اما برخلاف پروتکل ۱، قبل از ادامهٔ کار (آوردن بستهٔ بعدی و ارسال آن در قالب یک فریم) باید منتظر رسیدن فریم تصدیق دریافت از گیرنده بماند. نیازی نیست که لایهٔ پیوند داده فرستنده فریم دریافتی را بررسی کند، چون فقط یک احتمال وجود دارد: این فریم همیشه تصدیق دریافت گیرنده است.

تنها تفاوت *receiver2* با *receiver1* این است که بعد از تحویل بسته به لایهٔ شبکه و قبل از ورود به حالت انتظار، *receiver2* یک فریم تصدیق دریافت به فرستنده باز پس می‌فرستد. از آنجائیکه فقط خود فریم مهم است و نه محتویات آن، گیرنده هیچ داده‌ای در فیلد *info* این فریم قرار نمی‌دهد.

۳-۳-۳ پروتکل یکطرفه برای کانال‌های نویزدار

اکنون به یک حالت واقعی‌تر می‌پردازیم: کانالهایی که نویز دارند. فریمها می‌توانند با خطا به مقصد برسند، و یا بکلی گم شده و اصلاً به مقصد نرسند. با این حال، فرض می‌کنیم که اگر فریمی با خطا به مقصد رسید، سخت‌افزار لایهٔ فیزیکی با محاسبهٔ جمع تطبیقی متوجه خطا می‌شود. پروتکل ما در یک حالت به اشتباه عمل خواهد کرد: خطای رخ داده آنقدر شدید باشد که جمع تطبیقی تصادفاً درست از کار در آید (اتفاقی که بسیار نامحتمل است).

در نگاه اول با یک تغییر کوچک در پروتکل ۲ (اضافه کردن یک تایمر) می‌توان آن را با وضعیت جدید تطبیق داد. فرستنده می‌تواند در هر زمانی یک فریم بفرستد، ولی گیرنده فقط وقتی فریم تصدیق دریافت را برمی‌گرداند که این فریم را بدرستی دریافت و پردازش کرده باشد. اگر فریم ناقص به مقصد برسد، دور انداخته خواهد شد. بعد از مدتی تایمر فرستنده به انتها می‌رسد، و چون هنوز تصدیق دریافت گیرنده را نگرفته، مجدداً اقدام به ارسال فریم می‌کند. این ماجرا تا زمانی که فریم به سلامت به مقصد برسد، تکرار خواهد شد.

اما طرح بالا یک مشکل اساسی دارد. قبل از خواندن ادامهٔ کتاب، کمی فکر کنید و ببینید می‌توانید متوجه اشکال آن شوید.

برای درک مشکل، بیاد بیاورید که وظیفهٔ برقراری یک کانال ارتباطی عاری از خطا بین لایه‌های شبکه بر عهدهٔ لایهٔ پیوند داده است. لایهٔ شبکهٔ ماشین A یک سری بسته به لایهٔ پیوند داده می‌دهد، و باید مطمئن باشد که این بسته‌ها به همان ترتیبی که ارسال شده‌اند بدست لایهٔ شبکهٔ ماشین B خواهند رسید. بویژه، لایهٔ شبکهٔ ماشین B هیچ راهی ندارد تا بفهمد که یک بسته گم شده یا تکراریست. به همین دلیل لایهٔ پیوند داده ماشین B باید تضمین کند که هیچ بسته‌ای گم نمی‌شود، و یا تکراری نیست.

سناریوی زیر را در نظر بگیرید:

۱. لایهٔ شبکهٔ ماشین A بستهٔ ۱ را به لایهٔ پیوند داده می‌دهد. این بسته صحیح و سالم به لایهٔ پیوند داده ماشین B می‌رسد، و تحویل لایهٔ شبکه می‌شود. ماشین B یک فریم تصدیق دریافت به A می‌فرستد.

۲. فریم تصدیق دریافت ماشین B در راه از بین می‌رود، و هرگز به A نمی‌رسد. اگر فقط فریمهای داده گم می‌شدند و این اتفاق برای فریمهای کنترلی نمی‌افتاد، زندگی چقدر شیرین‌تر بود! ولی متأسفانه کانالهای مخابراتی اهل تبعیض نیستند!

۳. تایمر لایهٔ پیوند داده A به انتها می‌رسد، و چون هیچ فریم تصدیق دریافتی بدستش نرسیده، (باشتباه) تصور می‌کند که فریم به مقصد نرسیده (یا خراب شده)، پس آنرا دوباره می‌فرستد.

۴. فریم تکراری (در کمال صحت و سلامت) به لایهٔ پیوند داده B می‌رسد، و این لایه هم (بی خبر از همه جا) آنرا به لایهٔ شبکه تحویل می‌دهد. تصور کنید که اگر A در حال ارسال یک فایل به B باشد، تکراری بودن بخشی از آن چه فاجعه‌ای ببار خواهد آورد. همانطور که می‌بینید، پروتکل ما یک شکست کامل است.

چیزی که ما به آن احتیاج داریم، وسیله‌ایست که بتوان فریمهای تکراری را از فریمهایی که برای اولین بار دریافت می‌شوند، تشخیص داد. راه حل واضح این مشکل آن است که فرستنده یک **شمارهٔ ترتیبی** (sequence number) در سرآیند فریمهایی که می‌فرستد، قرار دهد. گیرنده می‌تواند این شماره را چک کرده، و فریمهای تکراری را دور بیندازد.

از آنجائیکه سرآیند یک فریم باید حتی‌الامکان کوچک باشد، سؤالی که پیش می‌آید اینست که: «حداقل تعداد بیتهای لازم برای فیلد شمارهٔ ترتیبی چندتاست؟» در پروتکل ما تنها ابهام در فریم m و فریم بعدی آن یعنی $m + 1$ است. اگر فریم m خراب شود یا از بین برود، گیرنده فریم تصدیق دریافت آنرا برنمی‌گرداند، پس فرستنده سعی می‌کند آنرا دوباره بفرستد. همین که این فریم سالم به مقصد رسید، گیرنده فریم تصدیق دریافت را به فرستنده پس می‌فرستد. همین جاست که مشکل بروز می‌کند: اگر فریم تصدیق دریافت صحیح و سالم به فرستنده بفرستد،

فرستنده فریم بعدی (یعنی $m + 1$) را می‌فرستد، در غیر اینصورت فریم m را خواهد فرستاد.

برای ارسال فریم $m + 2$، فرستنده باید قبلاً تصدیق دریافت فریم $m + 1$ را گرفته باشد. اما این بدان معناست که فریم m به سلامت به مقصد رسیده و تصدیق دریافت آن هم بدرستی به فرستنده برگشت داده شده است (چون در غیر اینصورت فرستنده فریم $m + 1$ را هم نمی‌فرستاد، چه رسد به فریم $m + 2$).

بنابراین، تنها ابهامی که می‌تواند وجود داشته باشد، بین یک فریم و فریم بعدی آن است. برای تشخیص این دو هم یک شمارهٔ ترتیبی یک‌بیتی (0 یا 1) کافیست. بعبارت دیگر، در هر لحظه گیرنده باید بدنبال شمارهٔ بعدی باشد. اگر فریمی با شمارهٔ اشتباه دریافت شد، گیرنده آنرا تکراری تلقی کرده و دور می‌اندازد. اما اگر شمارهٔ ترتیبی فریم درست بود، به لایهٔ شبکه تحویل داده می‌شود. با این توصیف فیلد شمارهٔ ترتیبی باید در مدول 2 افزایش داده شود (بعبارت دیگر، 1 به 0 تبدیل می‌شود، و 0 به 1).

پروتکلی با این مشخصات را در شکل ۳-۱۲ ملاحظه می‌کنید. به پروتکلهایی که فرستنده برای ارسال فریم بایستی منتظر یک تصدیق دریافت مثبت بماند، PAR (**تصدیق دریافت مثبت با ارسال مجدد** – Positive Acknowledgement with Retransmission) یا ARQ (**درخواست تکرار خودکار** – Automatic Repeat reQuest) نیز گفته می‌شود. این پروتکل هم، مانند پروتکل ۲، فقط در یک جهت داده می‌فرستد.

تفاوت پروتکل ۳ با دو تای قبلی اینست که، روالهای فرستنده و گیرنده متغیری دارند که مقدار آن حتی در زمانی که لایهٔ پیوند داده به حالت انتظار می‌رود، دست نخورده باقی می‌ماند. فرستنده باید شمارهٔ ترتیبی فریم بعدی که می‌خواهد بفرستد، را بداند: *next_frame_to_send* ؛ و گیرنده هم باید شمارهٔ ترتیبی فریم بعدی که باید منتظر آن باشد، را بداند: *frame_expected* .

```
/* Protocol 3 (par) allows unidirectional data flow over an unreliable channel. */
#define MAX_SEQ 1                                   /* must be 1 for protocol 3 */
typedef enum  {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"
void sender3(void)
{
  seq_nr next_frame_to_send;                        /* seq number of next outgoing frame
*/
  frame s;                                          /* scratch variable */
  packet buffer;                                    /* buffer for an outbound packet */
  event_type event;
  next_frame_to_send = 0;                           /*  initialize  outbound  sequence
numbers */
  from_network_layer(&buffer);                      /* fetch first packet */
  while (true) {
      s.info = buffer;                              /* construct a frame for transmission
*/
      s.seq = next_frame_to_send;                   /* insert sequence number in frame
*/
      to_physical_layer(&s);                        /* send it on its way */
      start_timer(s.seq);                           /* if answer takes too long, time out
```

```
*/
        wait_for_event(&event);                 /* frame_arrival, cksum_err, timeout
*/
        if (event == frame_arrival) {
            from_physical_layer(&s);            /* get the acknowledgement */
            if (s.ack == next_frame_to_send) {
                stop_timer(s.ack);              /* turn the timer off */
                from_network_layer(&buffer);    /* get the next one to send */
                inc(next_frame_to_send);        /* invert next_frame_to_send */
            }
        }
    }
}

void receiver3(void)
{
  seq_nr frame_expected;
  frame r, s;
  event_type event;
  frame_expected = 0;
  while (true) {
        wait_for_event(&event);                 /*      possibilities:      frame_arrival,
cksum_err */
        if (event == frame_arrival) {           /* a valid frame has arrived. */
            from_physical_layer(&r);            /* go get the newly arrived frame
*/
            if (r.seq == frame_expected) {      /* this is what we have been
waiting for. */
                to_network_layer(&r.info);      /* pass the data to the network
layer */
                inc(frame_expected);            /* next time expect the other
sequence nr */
            }
            s.ack = 1 - frame_expected;         /* tell which frame is being acked
*/
            to_physical_layer(&s);              /* send acknowledgement */
        }
    }
}
```

شکل ۳-۱۲. پروتکل تصدیق دریافت مثبت با ارسال مجدد (PAR) .

فرستنده، بعد از ارسال فریم، تایمر را راه می‌اندازد. اگر تایمر از قبل در حال اجرا باشد، این کار آنرا ریست کرده و آمادهٔ کار بعدی می‌کند. فاصلهٔ زمانی تایمر باید بگونه‌ای انتخاب شود که وقت کافی برای سه رویداد بدست دهد: رسیدن فریم به گیرنده، پردازش آن در گیرنده (در بدترین حالت)، و برگشت فریم تصدیق دریافت به فرستنده. فقط پس از سپری شدن این زمان است که می‌توان مطمئن شد فریم (یا تصدیق دریافت آن) به مقصد نرسیده، و فرستنده باید دوباره آنرا بفرستد. اگر فاصلهٔ زمانی تایمر کم انتخاب شود، تعداد دفعاتی که فرستنده فریم تکراری می‌فرستد افزایش یافته، و (با اینکه این کار تأثیر منفی روی گیرنده ندارد) به کارایی سیستم لطمه می‌زند.
بعد از ارسال فریم و راه انداختن تایمر، فرستنده منتظر یک اتفاق هیجان‌انگیز می‌ماند. البته فقط سه احتمال برای چنین اتفاقی وجود دارد: فریم تصدیق دریافت صحیح و سالم از راه برسد، فریم تصدیق دریافت با خطا وارد شود، تایمر منقضی شود (زمان مشخص شده به انتها برسد). در حالت اول، فرستنده بستهٔ دیگری از لایهٔ شبکه گرفته، و در بافر خود (*buffer*) قرار می‌دهد. سپس، شمارهٔ ترتیبی فریم (*next_frame_to_send*) را بالا می‌برد. اما در دو حالت دیگر (خراب شدن فریم تصدیق دریافت یا نرسیدن آن)، بافر و شمارهٔ ترتیبی هیچکدام تغییری نمی‌کند، بنابراین در هیچ حالتی فریم تکراری فرستاده نخواهد شد.
وقتی یک فریم به گیرنده می‌رسد، گیرنده شمارهٔ ترتیبی آن را چک می‌کند، تا از تکراری نبودن آن مطمئن شود. این فریم فقط در صورت تکراری نبودن به لایهٔ شبکه تحویل داده می‌شود. بدین ترتیب، فریمهای تکراری و خراب به لایهٔ شبکه نخواهند رسید.

۳-٤ پروتکل‌های پنجرهٔ لغزنده

در پروتکلهای قبل، فریمهای داده فقط در یک جهت ارسال می‌شدند. اما در عمل باید بتوانیم در هر دو جهت انتقال داده داشته باشیم. یکی از راههای داشتن یک کانال **دوطرفهٔ همزمان** (full-duplex) استفاده از دو کانال **یکطرفه** (simplex) در دو جهت مخالف است، که هر کدام فقط در یک جهت داده می‌فرستند (و البته فریمهای تصدیق دریافت می‌کنند). اما این روش چیزی جز اتلاف پهنای باند (و پول) نیست.
روش بهتر استفاده از یک کانال واحد برای ارسال داده در دو جهت است (مگر نه اینکه در پروتکلهای ۲ و ۳ در هر دو جهت فریم ارسال کردیم). از آنجائیکه در این مدل، فریمهای داده و تصدیق دریافت در هر دو جهت می‌توانند فرستاده شوند، باید کاری کنیم که گیرنده بتواند آنها را از یکدیگر تشخیص دهد. برای این منظور می‌توانیم از فیلد *info* در سرآیند فریمها استفاده کنیم.
با اینکه ترکیب فریمهای داده و تصدیق دریافت روی یک مدار واحد (بجای دو مدار جداگانه) یک قدم به جلو محسوب می‌شود، اما باز هم می‌توان کارایی سیستم را بهبود بخشید. وقتی گیرنده یک فریم داده دریافت می‌کند، بجای اینکه بلافاصله یک فرم کنترلی پس بفرستد، منتظر می‌ماند تا بستهٔ بعدی را برای ارسال از لایهٔ شبکه بگیرد. تصدیق دریافت فریم قبلی در فیلد *ack* فریم داده‌ای که اکنون می‌خواهد فرستاده شود، قرار داده می‌شود، و در واقع فریم تصدیق دریافت از فریم داده **سواری مجانی** (piggyback) می‌گیرد (و به همین نام هم خوانده می‌شود).
یکی از مزایای تکنیک سواری مجانی نسبت به ارسال مستقل فریمهای تصدیق دریافت، استفاده بهینه‌تر از پهنای باند موجود است: فیلد *ack* فقط چند بیت از سرآیند را اشغال می‌کند، در حالیکه یک فریم مستقل برای خود سرآیند، جمع تطبیقی، و تصدیق دریافت دارد. علاوه بر آن، هر چه تعداد فریمهایی که در یک جهت فرستاده می‌شوند کمتر باشد، گیرنده بهتر می‌تواند به کارهای دیگرش (پردازش فریمهای رسیده و خالی کردن بافرها) برسد، و این هم به بهبود کارایی سیستم کمک می‌کند. در پروتکلی که در این قسمت می نویسیم، فیلد سواری مجانی فقط یک بیت به سرآیند فریم اضافه می‌کند (و بندرت پیش می‌آید که مقدار آن از چند بیت بیشتر شود).

اما سواری مجانی هم خالی از اشکال نیست. برای مثال، در این حالت لایهٔ پیوند داده گیرنده چقدر باید منتظر بسته از لایهٔ شبکهٔ خود شود؟ اگر خیلی منتظر بماند، تایمر فرستنده به انتها رسیده و فریم را تکرار می‌کند، که این نقض غرض (از ارسال فریم تصدیق دریافت) است. اگر لایهٔ پیوند داده قدرت پیشگویی داشت، می‌توانست زمان دریافت بستهٔ بعدی از لایهٔ شبکه را پیشگویی کند، و آنوقت می‌توانست تصمیم بگیرد که منتظر این بسته بماند یا بلافاصله فریم تصدیق دریافت را به فرستنده بفرستد. اما متأسفانه لایهٔ پیوند داده نمی‌تواند آینده را پیشگویی کند، پس باید روش ساده‌تری (انتظار بمدت ثابت، مثلاً چند میلی‌ثانیه) پیدا کنیم. اگر در این فاصله بسته‌ای از لایهٔ شبکه رسید، لایهٔ پیوند داده تصدیق دریافت را سوار آن می‌کند؛ در غیر اینصورت یک فریم مستقل تصدیق دریافت به سمت فرستنده ارسال می‌کند.

پروتکلهایی که در این قسمت خواهید دید، به کلاس پروتکلهای **پنجرهٔ لغزنده** (sliding window) تعلق دارند، که فقط از نظر کارایی، پیچیدگی و بافر با هم تفاوت دارند. در این پروتکلها، مانند سایر پروتکلهای پنجرهٔ لغزنده، هر فریم خروجی یک شمارهٔ ترتیبی (از 0 تا یک حداکثر) دارد. حداکثر شمارهٔ فریمها معمولاً $2^n - 1$ است، بنابراین در یک فیلد n-بیتی بخوبی جا می‌شود. پروتکل پنجرهٔ لغزندهٔ توقف-انتظار از $n = 1$ استفاده می‌کرد، اما در پروتکلهای دیگر این عدد می‌تواند بیشتر باشد.

ایدهٔ اصلی در تمام پروتکلهای پنجرهٔ لغزنده این است که، فرستنده در هر لحظه از زمان لیستی از شماره‌های ترتیبی متناظر با فریمهایی که می‌تواند ارسال کند، در اختیار دارد. اصطلاحاً گفته می‌شود که این فریمها در **پنجرهٔ ارسال** (sending window) قرار دارند. گیرنده هم یک **پنجرهٔ دریافت** (receiving window) دارد که متناظر است با فریمهایی که مجاز به دریافت آنهاست. الزامی نیست که پنجرهٔ ارسال و پنجرهٔ دریافت حد پائین و بالای مشابه داشته باشند، یا حتی هم‌اندازه باشند. در برخی پروتکلها اندازهٔ این پنجره‌ها ثابت است، ولی در پروتکلهای دیگر می‌تواند کوچک یا بزرگ شود.

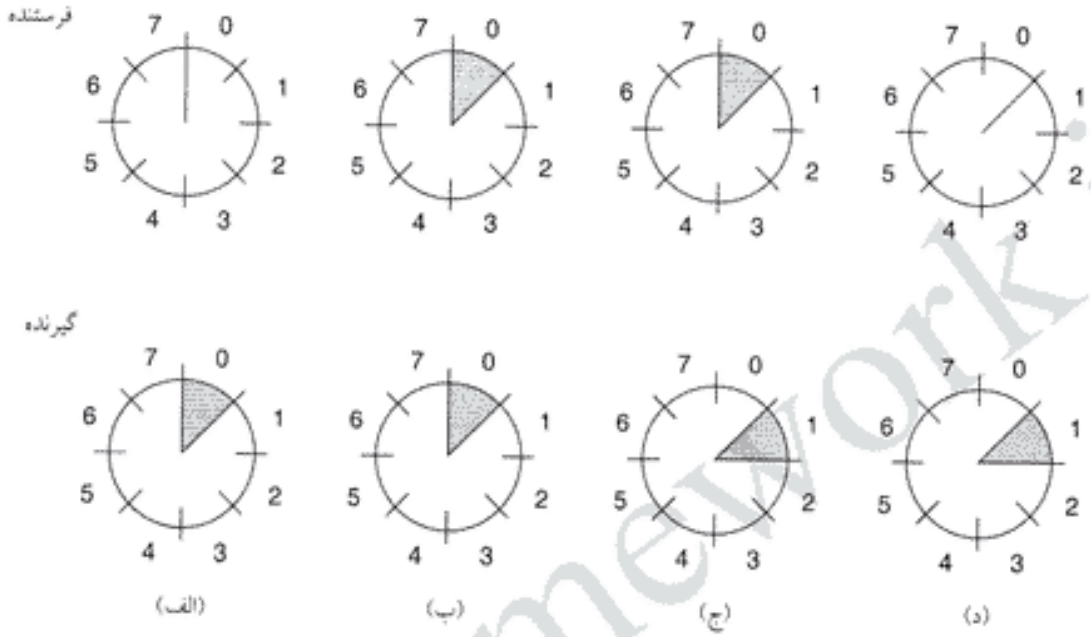
با اینکه این پروتکلها آزادی عمل بیشتری به لایهٔ پیوند داده در ارسال و دریافت فریمها می‌دهند، لازم است مجدداً تأکید کنیم که تحویل بسته‌ها به لایهٔ شبکهٔ مقصد باید با همان ترتیبی صورت گیرد که در ماشین مبدأ تحویل لایهٔ پیوند داده شده‌اند. (و یک بار دیگر خاطرنشان می‌کنیم که، لایهٔ فیزیکی یک کانال ارتباطی ساده است که فریمها را به همان ترتیبی که به آن داده شده، منتقل می‌کند.)

شماره‌های موجود در پنجرهٔ ارسال شمارهٔ فریمهائیست که باید ارسال شوند، و یا ارسال شده‌اند ولی هنوز تصدیق دریافت آنها برنگشته است. وقتی یک بستهٔ جدید از لایهٔ شبکه می‌رسد، لایهٔ پیوند داده بالاترین شمارهٔ ترتیبی موجود را به آن می‌دهد، و لبهٔ بالایی پنجرهٔ ارسال را یکی زیاد می‌کند. وقتی یک تصدیق دریافت وارد شد،لبهٔ پائینی پنجرهٔ ارسال نیز بالا برده می‌شود. بدین ترتیب پنجرهٔ ارسال همیشه شامل لیست فریمهائیست که دریافت آنها هنوز توسط گیرنده تصدیق نشده است. در شکل ۳-۱۳ یک نمونه را ملاحظه می‌کنید.

از آنجائیکه امکان گم یا خراب شدن فریمهایی که در حال حاضر در پنجرهٔ ارسال قرار دارند، همیشه وجود دارد، فرستنده باید آنها را برای ارسال مجدد (احتمالی) در حافظه نگه دارد. بنابراین اگر حداکثر اندازهٔ این پنجره n باشد، فرستنده باید بافری باندازهٔ n فریم تصدیق نشده داشته باشد. اگر پنجرهٔ ارسال از حداکثر پیش‌بینی شده بزرگتر شود، لایهٔ پیوند داده باید گرفتن بسته از لایهٔ شبکه را تا زمان آزاد شدن بافر متوقف کند.

پنجرهٔ دریافت در گیرنده متناظر است با فریمهایی که گیرنده مجاز به دریافت آنهاست. هر فریمی که خارج از این پنجره قرار گیرد، بدون هیچ توضیحی دور انداخته خواهد شد. وقتی فریمی که شمارهٔ ترتیبی آن معادل لبهٔ پائین پنجرهٔ دریافت است، از راه می‌رسد، به لایهٔ شبکه تحویل شده و پس از ایجاد تصدیق دریافت آن، پنجرهٔ دریافت یک واحد می‌چرخد. بر خلاف پنجرهٔ فرستنده، پنجرهٔ گیرنده همیشه به همان اندازهٔ اولیه می‌ماند. توجه

کنید که پنجرهٔ دریافت 1 بمعنای اینست که لایهٔ پیوند داده فریمها را فقط به ترتیب می‌پذیرد، ولی در پنجره‌های بزرگتر الزاماً چنین نیست (با این حال، لایهٔ شبکه همیشه داده‌ها را به ترتیب صحیح تحویل لایهٔ پیوند داده می‌دهد).



شکل ۳-۱۳. یک پنجرهٔ لغزندهٔ یک واحدی، با شمارهٔ ترتیبی ۳-بیتی. (الف) در شروع کار. (ب) بعد از ارسال اولین فریم. (ج) بعد از آنکه اولین فریم دریافت شد. (د) بعد از آنکه فرستنده اولین تصدیق دریافت را گرفت.

در شکل ۳-۱۳ حداکثر اندازهٔ پنجره 1 است. در لحظهٔ اول لبه‌های پائین و بالای پنجرهٔ ارسال یکی هستند، ولی با گذشت زمان موقعیت آنها طبق شکل تغییر می‌کند.

۳-٤-۱ پروتکل پنجرهٔ لغزندهٔ 1-بیتی

قبل از پرداختن به حالت کلی، اجازه دهید ابتدا پروتکلی با پنجرهٔ لغزندهٔ 1-بیتی را مورد بررسی قرار دهیم. در واقع این پروتکل نوعی پروتکل توقف-انتظار است، چون فرستنده قبل از گرفتن تصدیق دریافت فریم فرستاده شده، فریم بعدی را ارسال نخواهد کرد.

در شکل ۳-۱۴ پروتکل پنجرهٔ لغزندهٔ 1-بیتی را ملاحظه می‌کنید. مانند پروتکلهای قبلی، این پروتکل هم با تعریف متغیرها شروع می‌شود. متغیر *next_frame_to_send* فریم بعدیست که فرستنده باید بفرستد. در طرف مقابل هم، متغیر *frame_expected* فریمی را نشان می‌دهد که گیرنده منتظر آن است. در هر دو طرف، تنها حالتهای مجاز فقط 0 یا 1 است.

```
/* Protocol 4 (sliding window) is bidirectional. */
#define MAX_SEQ 1                                   /* must be 1 for protocol 4 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"
void protocol4 (void)
```

```
    {
      seq_nr next_frame_to_send;                    /* 0 or 1 only */
      seq_nr frame_expected;                        /* 0 or 1 only */
      frame r, s;                                   /* scratch variables */
      packet buffer;                                /* current packet being sent */
      event_type event;
      next_frame_to_send = 0;                       /* next frame on the outbound
stream */
      frame_expected = 0;                           /* frame expected next */
      from_network_layer(&buffer);                  /* fetch a packet from the network
layer */
      s.info = buffer;                              /* prepare to send the initial frame */
      s.seq = next_frame_to_send;                   /* insert sequence number into
frame */
      s.ack = 1 - frame_expected;                   /* piggybacked ack */
      to_physical_layer(&s);                        /* transmit the frame */
      start_timer(s.seq);                           /* start the timer running */
      while (true) {
          wait_for_event(&event);                   /* frame_arrival, cksum_err, or
timeout */
          if (event == frame_arrival) {             /* a frame has arrived
undamaged. */
              from_physical_layer(&r);              /* go get it */
              if (r.seq == frame_expected) {        /* handle inbound frame
stream. */
                  to_network_layer(&r.info);        /* pass packet to network
layer */
                  inc(frame_expected);              /* invert seq number expected
next */
              }
              if (r.ack == next_frame_to_send) {    /* handle outbound frame
stream. */
                  stop_timer(r.ack);                /* turn the timer off */
                  from_network_layer(&buffer);      /* fetch new pkt from network
layer */
                  inc(next_frame_to_send);          /* invert sender's sequence
number */
              }
          }
          s.info = buffer;                          /* construct outbound frame */
          s.seq = next_frame_to_send;               /* insert sequence number into it
```

```
*/
        s.ack = 1 - frame_expected;          /* seq number of last received
frame */
        to_physical_layer(&s);               /* transmit a frame */
        start_timer(s.seq);                  /* start the timer running */
    }
  }
```

شکل ۳-۱۴. پروتکل پنجرهٔ لغزندهٔ 1-بیتی.

در شرایط عادی، یکی از دو طرف پیش دستی کرده و اولین فریم را می‌فرستد. بعبارت دیگر، فقط یکی از دو لایهٔ پیوند داده روالهای *to_physical_layer* و *start_timer* را خارج از حلقهٔ اصلی برنامه اجرا می‌کند. اگر در پیشامدی نادر هر دو طرف بطور همزمان شروع به ارسال اولین فریم کنند، وضعیت عجیبی پیش می‌آید، که بعداً آنرا توضیح خواهیم داد. ماشین شروع‌کننده اولین بسته را از لایهٔ شبکه گرفته، یک فریم از آن می‌سازد، و سپس ارسال می‌کند. وقتی این فریم (یا هر فریم دیگری) به طرف مقابل رسید، لایهٔ پیوند داده گیرنده چک می‌کند که تکراری نباشد (درست مثل پروتکل ۳). اگر این همان فریم موردنظر باشد، به لایهٔ شبکه تحویل شده و پنجرهٔ دریافت یک واحد به جلو لغزانده می‌شود.

فیلد تصدیق دریافت حاوی شمارهٔ آخرین فریمیست که بدون خطا دریافت شده است. اگر این شماره با شمارهٔ فریمی که فرستنده در صدد ارسال آن است یکی باشد، فرستنده می‌فهمد که دیگر نیازی به فریم داخل بافر ندارد و می‌تواند بستهٔ بعدی را از لایهٔ شبکه بگیرد. اگر شماره‌ها یکی نباشند، فرستنده باید به ارسال همان فریم قبلی ادامه دهد. وقتی یک فریم دریافت شود، فریمی نیز پس فرستاده می‌شود.

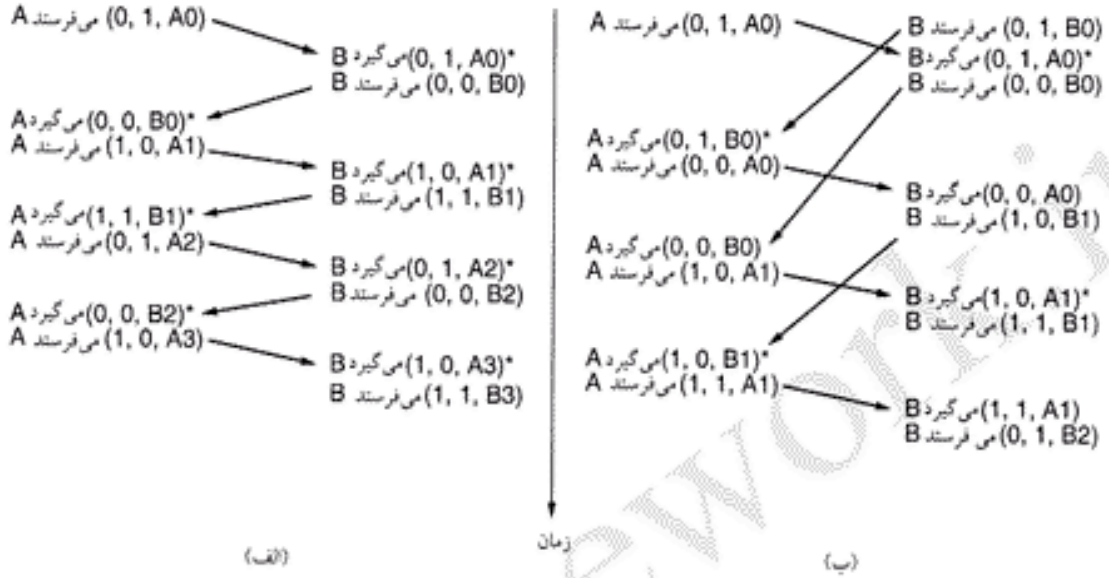
حال اجازه دهید ببینیم پروتکل ۴ در شرایط غیرعادی چگونه رفتار می‌کند. فرض کنید ماشین A درصدد ارسال فریم 0 به کامپیوتر B است، و در همان زمان B نیز تصمیم می‌گیرد فریم 0 خود را به A بفرستد. در ضمن فرض می‌کنیم که A مسابقه را زودتر شروع می‌کند، ولی فاصلهٔ زمانی تایمر آن بسیار کوتاه است. در نتیجه، ماشین A پشت سر هم فریمهای یکسان، با $seq = 0$ و $ack = 1$، به B می‌فرستد.

وقتی اولین فریم به کامپیوتر B رسید، پذیرفته شده و *frame_expected* به 1 ست می‌شود؛ تمام فریمهای بعدی رد خواهند شد، چون اینک B در انتظار فریمی با شمارهٔ ترتیبی 1 است نه 0 . علاوه بر آن، چون تمام در فریمهای تکراری $ack = 1$ و B همچنان منتظر تصدیق دریافت فریم 0 است، ماشین B گرفتن بسته از لایهٔ شبکهٔ خود را متوقف خواهد کرد.

بعد از آن که تمام فریمهای تکراری به B رسیدند، B یک فریم با $seq = 0$ و $ack = 0$ به A می‌فرستد. بالاخره یکی از این فریمها سالم به A می‌رسد، و باعث می‌شود تا A ارسال فریم بعدی را شروع کند. همانطور که می‌بینید، در هیچ شرایطی بستهٔ تکراری به لایهٔ شبکه نمی‌رسد، بسته‌ای گم نمی‌شود، و سیستم قفل نمی‌کند.

اما اگر هر دو طرف در یک لحظه شروع به ارسال اولین فریم کنند، وضعیت عجیبی پیش می‌آید. این مشکل سنکرون شدن را در شکل ۳-۱۵ (ب) مشاهده می‌کنید. (در شکل ۳-۱۵ الف برای مقایسه عملکرد عادی پروتکل ۴ نشان داده شده است.) اگر B قبل از ارسال فریمهای خود منتظر دریافت اولین فریم A بماند، هیچ مشکلی پیش نمی‌آید و تمام بسته‌ها براحتی پذیرفته می‌شوند (شکل الف). ولی اگر هر دو با هم مخابرهٔ اولین فریم را شروع کنند، این فریمها با یکدیگر تصادم کرده، و حالت (ب) پیش می‌آید. در (الف) دریافت هر فریم باعث گرفتن یک بسته از لایهٔ شبکه می‌شود، و هیچ فریم تکراری هم وجود ندارد. اما در (ب) با وجود اینکه هیچ خطایی هم در

کانال وجود ندارد، نصف فریمها تکراری هستند. این وضعیت هنگامی که تایمر یکی از دو طرف بیش از حد کوتاه باشد، نیز پیش می‌آید (حتی اگر دو طرف همزمان شروع به مخابرهٔ فریمهای خود نکرده باشند). در چنین وضعیتی حتی امکان دارد برخی از فریمها سه یا چهار بار تکرار شوند.



شکل ۳-۱۵. دو سناریوی پروتکل ۴. (الف) حالت عادی. (ب) حالت غیرعادی. اعداد داخل پرانتز از چپ به راست عبارتند از: seq ، ack و شمارهٔ بسته. بسته‌هایی که پذیرفته و به لایهٔ شبکه تحویل می‌شوند، با * مشخص شده‌اند.

۳-٤-۲ پروتکل «N تا به عقب برگرد»

تا اینجا بطور ضمنی فرض کرده بودیم که زمان رسیدن فریم به مقصد بعلاوهٔ زمان برگشت فریم تصدیق ناچیز است. اما گاهی این فرض آشکارا نادرست است. در چنین مواردی زمان طولانی رفت و برگشت فریم می‌تواند تأثیر چشمگیری روی کارایی مصرف پهنای باند داشته باشد. بعنوان مثال، یک کانال ماهواره‌ای با پهنای باند 50 kbps و زمان تأخیر رفت و برگشت 500 msec را در نظر بگیرید. فرض کنید می‌خواهیم با این لینک ماهواره‌ای و با استفاده از پروتکل ۴ فریمهای 1000 بیتی ارسال کنیم. در لحظهٔ $t = 0$ فرستنده مخابرهٔ اولین فریم را شروع می‌کند. و در $t = 20$ msec کار ارسال فریم به پایان می‌رسد. اما در بهترین شرایط (و با فرض اینکه در گیرنده نیز هیچ تأخیری وجود ندارد)، تا $t = 270$ msec این فریم هنوز به گیرنده نرسیده، و تا $t = 520$ msec نیز مسلماً فریم تصدیق دریافت به دست فرستنده نخواهد رسید. این بدان معناست که فرستنده در 500/520 یا 96% زمان باید متوقف بماند، و نمی‌تواند چیزی بفرستد؛ بعبارت دیگر، از فقط 4% پهنای باند استفاده می‌کند. پیداست که ترکیب تأخیر طولانی در کانال، پهنای باند زیاد، و فریمهای کوچک چیزی جز اتلاف وحشتناک منابع نیست.

مشکلی که در بالا دیدید از آنجا ناشی می‌شد که فرستنده برای ارسال یک فریم باید منتظر تصدیق دریافت فریم قبلی بماند. اما اگر این قید را برداریم، می‌توانیم به کارایی بهتری دست پیدا کنیم. در واقع بجای 1 فریم، معمولاً فرستنده می‌تواند w فریم بفرستد و پس از آن منتظر رسیدن فریمهای تصدیق بماند. اگر w طوری انتخاب شود که فرستنده در تمام مدت زمان تأخیر رفت و برگشت در حال ارسال فریم باشد، می‌تواند بدون مشکل از تمام پهنای باند استفاده کند. در مثال بالا w باید حداقل 26 باشد. فرستنده مانند قبل ارسال اولین فریم را در $t = 0$ شروع می‌کند، و زمانی که فریم 26 را می‌فرستد ($t = 520$ msec)، تصدیق فریم 0 را دریافت خواهد کرد. از آن به

بعد نیز فریمهای تصدیق دریافت هر 20 msec از راه می‌رسند، و فرستنده می‌تواند بطور پیوسته به ارسال فریمها ادامه دهد. در تمام زمانها فرستنده 25 یا 26 فریم در بافر خود دارد که هنوز تصدیق دریافت آنها را نگرفته است. به بیان دیگر، اندازهٔ پنجره ارسال حداکثر 26 است.

پنجرهٔ ارسال بزرگ فقط زمانی لازم می‌شود که حاصلضرب **پهنای باند** × **تأخیر رفت و برگشت** عددی بزرگ باشد. اگر پهنای باند زیاد باشد، حتی با تأخیر کم نیز فرستنده بسرعت پنجرهٔ ارسال را پُر می‌کند. اگر تأخیر رفت و برگشت زیاد باشد (مانند کانالهای ماهواره‌ای GEO)، حتی در پهنای باند متوسط نیز پنجرهٔ ارسال بزودی پُر می‌شود. ظرفیت یک کانال اساساً با حاصلضرب این دو عامل (پهنای باند و تأخیر رفت و برگشت) تعیین می‌شود، و فرستنده برای رسیدن به حداکثر کارایی باید بتواند کانال را بدون وقفه پُر کند.

به این تکنیک **لوله‌کشی** (pipelining) گفته می‌شود. اگر ظرفیت کانال b bits/sec، اندازهٔ فریم l بیت، و تأخیر رفت و برگشت R sec باشد، زمان لازم برای ارسال هر فریم l/b sec خواهد بود. بعد از ارسال آخرین بیت یک فریم، و قبل از رسیدن این بیت به گیرنده، تأخیری بمیزان $R/2$ وجود دارد؛ بازگشت فریم تصدیق دریافت نیز با همین مقدار تأخیر همراه است (که کل تأخیر به R می‌رسد). با پروتکل توقف-انتظار، خط بمدت l/b کار کرده و سپس بمدت R بیکار می‌ماند، که در نتیجه

$$\text{بهرهٔ خط} = l / (l + bR)$$

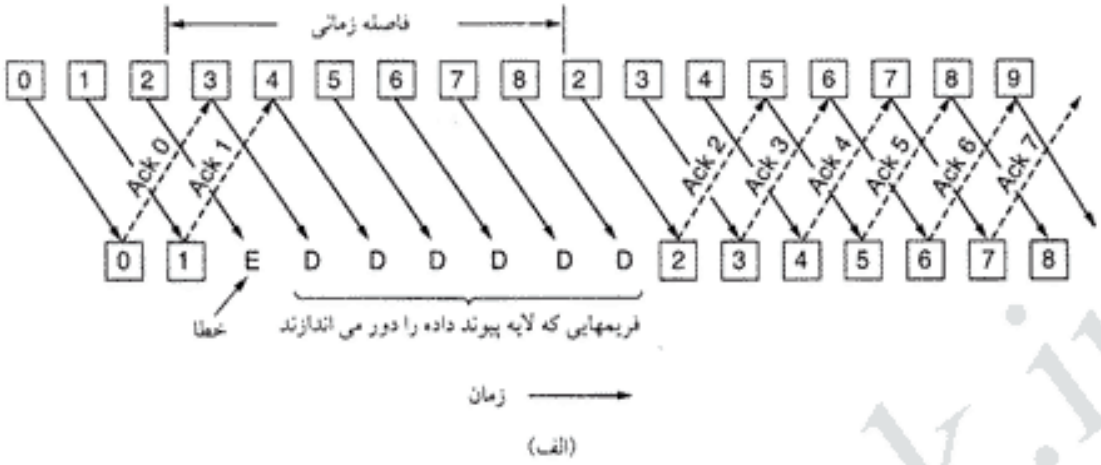
اگر $l < bR$، کارایی خط زیر 50% خواهد بود. از آنجائیکه تأخیر رفت و برگشت خطوط انتقال هرگز صفر نیست، با تکنیک لوله‌کشی می‌توان کاری کرد که خط همیشه مشغول باشد - اما اگر مقدار این تأخیر ناچیز باشد، ارزش پیچیدگی بیشتر پروتکل را ندارد.

لوله‌کشی فریمها در کانالهای غیرقابل‌اطمینان می‌تواند منجر به مشکلات جدی شود. اول اینکه، اگر یکی از فریمهای این صف طویل ناپدید یا خراب شود، چه خواهد شد؟ قبل از آنکه حتی فرستنده متوجه این خطا شود، تعداد زیادی از فریمهای بعدی به گیرنده رسیده‌اند. با رسیدن فریم خراب، گیرنده مسلماً آنرا دور می‌اندازد، اما با فریمهای سالم بعدی چه باید بکند؟ بیاد داشته باشید که لایهٔ پیوند داده باید بسته‌ها را با ترتیب صحیح به لایهٔ شبکه تحویل دهد. در شکل ۳-۱۶ این وضعیت (بروز خطا در **خط لوله** - pipeline) را ملاحظه می‌کنید. اجازه دهید آنرا دقیقتر بررسی کنیم.

برای مقابله با خطا در تکنیک لوله‌کشی دو رهیافت کلی وجود دارد. در رهیافت اول، که به «N تا به عقب برگرد» معروف است، گیرنده تمام فریمهای بعد از فریم خراب را دور می‌اندازد و هیچ تصدیقی برای آنها برنمی‌گرداند. این استراتژی معادل است با پنجرهٔ دریافتی باندازهٔ 1. بعبارت دیگر، لایهٔ پیوند داده در گیرنده هیچ فریمی غیر ازآن فریمی که باید به لایهٔ شبکه تحویل دهد، را قبول نمی‌کند. اگر پنجرهٔ ارسال فرستنده قبل از انقضای تایمر پُر شود، خط لوله شروع به خالی شدن خواهد کرد. پس از آن تایمر فرستنده به انتها رسیده و تمام فریمهای باقی مانده (از فریمی که خراب یا گم شده) را دوباره ارسال خواهد کرد. اگر نرخ خطا در خط زیاد باشد (مانند کانالهای بیسیم)، این رهیافت باعث اتلاف شدید پهنای باند خواهد شد.

در شکل ۳-۱۶ (الف) این حالت را ملاحظه می‌کنید. در این مثال فریمهای 0 و 1 سالم به مقصد رسیده‌اند، ولی فریم 2 خراب شده است. فرستنده تا زمانی که تایمر این فریم منقضی نشود، از خراب شدن آن مطلع نخواهد شد. پس از آنکه فرستنده فهمید فریم 2 سالم به مقصد نرسیده، برمی‌گردد و ارسال فریمها را از این فریم از سر می‌گیرد.

رهیافت دوم مقابله با خطا در تکنیک لوله‌کشی **تکرار انتخابی** (selective repeat) نام دارد. در این رهیافت، فریم خراب در گیرنده دور انداخته شده، ولی فریمهای سالم بعدی باقر می‌شوند. وقتی تایمر فریم معیوب منقضی



(الف)

(ب)

شکل ۳-۱۶. مقابله با خطا در خط لوله. تأثیر خطا وقتی که (الف) اندازهٔ پنجرهٔ دریافت گیرنده 1 است، و (ب) پنجرهٔ دریافت بزرگ است.

شد، فرستنده فقط همین فریم (که قدیمی‌ترین فریم تصدیق نشده است) را مجدداً ارسال می‌کند. اگر این فریم سالم به مقصد رسید، گیرنده این فریم و فریمهای بافر شده را بترتیب به لایهٔ شبکه تحویل می‌دهد. در رهیافت تکرار انتخابی، معمولاً گیرنده برای فریمهای خراب (فریمهایی که جمع تطبیقی آنها اشتباه است، یا خارج از نظم وارد می‌شوند) نیز یک فریم **تصدیق دریافت منفی** (negative acknowledgement) - که به فریم NAK معروف است - به فرستنده برمی‌گرداند. فریم NAK کارایی سیستم را به مقدار زیادی بهبود می‌بخشد، چون باعث می‌شود که فرستنده قبل از انقضای تایمر کار ارسال مجدد فریمهای از دست رفته را شروع کند.

در شکل ۳-۱۶ (ب) نیز فریمهای 0 و 1 سالم به مقصد رسیده‌اند، ولی فریم 2 خراب شده است. وقتی فریم 3 به گیرنده می‌رسد، لایهٔ پیوند داده متوجه می‌شود که یک فریم جا افتاده است، پس یک NAK برای فریم 2 به فرستنده فرستاده، و فریم 3 را در بافری که برای این منظور اختصاص یافته، ذخیره می‌کند. فریمهای 4 و 5 نیز پس از رسیدن به گیرنده، بجای تحویل به لایهٔ شبکه، بافر می‌شوند. با رسیدن NAK فریم 2 ، فرستنده بلافاصله این فریم را ارسال می‌کند، که با رسیدن آن به مقصد، گیرنده می‌تواند فریم 2 و فریمهای پس از آن (تا فریم 5) را به لایهٔ شبکه تحویل دهد (و تصدیق دریافت آنها را به فرستنده برگرداند). حتی اگر NAK فریم 2 در راه گم شود و به دست فرستنده نرسد، باز هم پس از انقضای تایمر، فرستنده نسبت به ارسال مجدد آن (و فقط همین یک فریم) اقدام خواهد کرد؛ که البته در این میان فقط وقت بیشتری تلف خواهد شد. در واقع، NAK فقط ارسال مجدد

فریمهای معیوب را تسریع می‌کند.

رهیافت تکرار انتخابی خاص پنجره‌های دریافت بزرگتر از 1 است. هر فریمی در داخل پنجرهٔ دریافت قرار داشته باشد، می‌تواند بافر شود تا فریمهای قبل از آن دریافت و به لایهٔ شبکه تحویل شوند. البته اگر پنجرهٔ دریافت خیلی بزرگ باشد، لایهٔ پیوند داده به حافظهٔ زیادی برای بافر کردن فریمها نیاز دارد.

این دو رهیافت داد و ستدی هستند بین پهنای باند و فضای بافر لایهٔ پیوند داده، که بسته به اهمیت هر یک از این منابع می‌توان یکی از رهیافت‌های «N تا به عقب برگرد» یا «تکرار انتخابی» را انتخاب کرد. در شکل ۳-۱۷ پروتکل لوله‌کشی با پنجرهٔ دریافت 1 را ملاحظه می‌کنید؛ در این پروتکل تمام فریمهای بعد از یک فریم معیوب دور انداخته می‌شوند. در این پروتکل برای اولین بار فرض نامحدود بودن بسته‌هایی که لایهٔ شبکه می‌تواند ارائه کند، را نیز کنار گذاشته‌ایم. در اینجا، وقتی لایهٔ شبکه آمادهٔ ارسال یک بسته است، رویداد *network_layer_ready* را تحریک می‌کند. با این حال، برای آنکه هیچوقت بیش از *MAX_SEQ* فریم وارد صف ارسال لایهٔ پیوند داده نشود، باید کاری کنیم که این لایه بتواند بطریقی مانع ارسال بسته‌های اضافی از لایهٔ شبکه شود - برای این منظور از روالهای کتابخانه‌ای *enable_network_layer* و *disable_network_layer* استفاده کرده‌ایم.

```
/* Protocol 5 (go back n) allows multiple outstanding frames. The sender may transmit up
   to MAX_SEQ frames without waiting for an ack. In addition, unlike in the previous
   protocols, the network layer is not assumed to have a new packet all the time. Instead,
the network layer causes a network_layer_ready event when there is a packet to send.
*/

#define MAX_SEQ 7                                   /* should be 2^n - 1 */
typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready} event_type;
#include "protocol.h"

static boolean between(seq_nr a, seq_nr b, seq_nr c)
{
/* Return true if a <= b < c circularly; false otherwise. */
  if (((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a)))
        return(true);
    else
        return(false);
}

static void send_data(seq_nr frame_nr, seq_nr frame_expected, packet buffer[])

{
/* Construct and send a data frame. */
  frame s;                                  /* scratch variable */

  s.info = buffer[frame_nr];                    /* insert packet into frame */
```

```
  s.seq = frame_nr;                          /* insert sequence number into frame */
  s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);       /* piggyback ack */
  to_physical_layer(&s);                     /* transmit the frame */
  start_timer(frame_nr);                        /* start the timer running */
}

void protocol5(void)
{
  seq_nr next_frame_to_send;                    /* MAX_SEQ > 1; used for outbound
stream */
  seq_nr ack_expected;                       /* oldest frame as yet unacknowledged */
  seq_nr frame_expected;                        /* next frame expected on inbound
stream */
  frame r;                                   /* scratch variable */
  packet buffer[MAX_SEQ + 1];                   /* buffers for the outbound stream */
  seq_nr nbuffered;                             /* # output buffers currently in use */
  seq_nr i;                                     /* used to index into the buffer array */
  event_type event;

  enable_network_layer();                       /* allow network_layer_ready events */
  ack_expected = 0;                             /* next ack expected inbound */
  next_frame_to_send = 0;                       /* next frame going out */
  frame_expected = 0;                           /* number of frame expected inbound */
  nbuffered = 0;                             /* initially no packets are buffered */
  while (true) {
     wait_for_event(&event);                    /* four possibilities: see event_type
above */

     switch(event) {
        case network_layer_ready:               /* the network layer has a packet to send
*/
             /* Accept, save, and transmit a new frame. */
             from_network_layer(&buffer[next_frame_to_send]); /* fetch new packet */
             nbuffered = nbuffered + 1;            /* expand the sender's window */
             send_data(next_frame_to_send, frame_expected, buffer); /* transmit the
frame */
             inc(next_frame_to_send);              /* advance sender's upper window
edge */
             break;

        case frame_arrival:                     /* a data or control frame has arrived */
```

```
            from_physical_layer(&r);            /* get incoming frame from physical
layer */

            if (r.seq == frame_expected) {
                /* Frames are accepted only in order. */
                to_network_layer(&r.info);      /* pass packet to network layer */
                inc(frame_expected);            /* advance lower edge of receiver's
window */
            }

            /* Ack n implies n - 1, n - 2, etc.  Check for this. */
            while (between(ack_expected, r.ack, next_frame_to_send)) {
                /* Handle piggybacked ack. */
                nbuffered = nbuffered - 1; /* one frame fewer buffered */
                stop_timer(ack_expected); /* frame arrived intact; stop timer */

                inc(ack_expected);          /* contract sender's window */
            }
            break;

        case cksum_err: break;              /* just ignore bad frames */

        case timeout:                       /* trouble; retransmit all outstanding
frames */
            next_frame_to_send = ack_expected;   /* start retransmitting here */
            for (i = 1; i <= nbuffered; i++) {
                send_data(next_frame_to_send, frame_expected, buffer); /* resend
1 frame */
                inc(next_frame_to_send);    /* prepare to send the next one */
            }

    }

    if (nbuffered < MAX_SEQ)
 ;)(reyal_krowten_elbane
    else
            disable_network_layer();
  }
}
```

شکل ۳-۱۷. پروتکل پنجرهٔ لغزندهٔ N تا به عقب برگرده.

توجه کنید که در هر لحظه نباید بیش از MAX_SEQ فریم (و نه $MAX\_SEQ + 1$) در صف ارسال وجود داشته باشد – حتی با وجود اینکه تعداد شماره‌های ترتیبی، در $MAX\_SEQ + 1$ است: $0, 1, 2, ..., MAX\_SEQ$ برای پی بردن به علت این محدودیت، سناریوی زیر را که در آن $MAX\_SEQ = 7$ نظر بگیرید:

۱. فرستنده فریمهای 0 تا 7 (هشت فریم) را ارسال می‌کند.
۲. پس از مدتی، تصدیق دریافت فریم 7 (با استفاده از تکنیک سواری‌مجانی) به فرستنده برمی‌گردد.
۳. فرستنده ۸ فریم بعدی را با همان شماره‌های ترتیبی 0 تا 7 ارسال می‌کند.
٤. حال فریم تصدیق دریافت دیگری برای فریم شمارهٔ 7 (به همان صورت سواری مجانی) از راه می‌رسد.

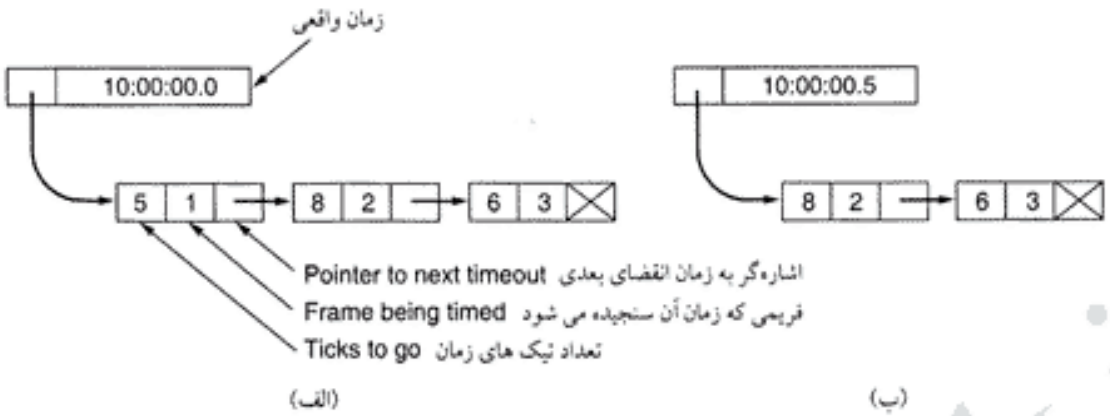
سؤال این است: آیا تمام ۸ فریمی که متعلق به دسته دوم بودند، سالم به مقصد رسیده‌اند، یا (با توجه به اینکه تمام فریمهای پس از یک فریم خراب دور انداخته می‌شوند) همگی از بین رفته‌اند؟ اگر دقت کرده باشید، در هر دو حالت گیرنده تصدیق دریافت فریم 7 را پس می‌فرستد، و فرستنده هم راهی برای تشخیص این موضوع ندارد. به همین دلیل حداکثر فریمهایی که در صف ارسال می‌ایستند، باید MAX_SEQ باشد.

با اینکه پروتکل ۵ فریمهای پس از یک فریم معیوب را بافر نمی‌کند، اما هنوز به مقداری بافر در سمت فرستنده نیاز داریم. از آنجائیکه فرستنده باید تمام فریمهای داخل پنجرهٔ ارسال را تا رسیدن تصدیق دریافت آنها نگه دارد (چون ممکنست لازم شود آنها را دوباره بفرستد)، باید فضای کافی برای بافر کردن این فریمها داشته باشد. در این پروتکل وقتی تصدیق دریافت فریم n از راه برسد، فریمهای $n - 1$، $n - 2$ و ... نیز بطور خودکار تصدیق شده محسوب می‌شوند. این ویژگی بویژه اگر فریمهای تصدیق قبلی در راه برگشت به فرستنده از بین بروند، اهمیت می‌یابد. با رسیدن هر فریم تصدیق دریافت، لایهٔ پیوند داده چک می‌کند که کدام بافرها را می‌تواند آزاد کند. وقتی بافر آزاد (و جایی در پنجرهٔ ارسال باز) شد، لایهٔ شبکه (که قبلاً متوقف شده) دوباره با رویداد *enable_network_layer* فعال می‌شود و می‌تواند بسته‌های بعدی را به لایهٔ پیوند داده بفرستد.

در پروتکل ۵ فرض کرده‌ایم که همیشه ترافیک برگشتی کافی برای سواری‌مجانی وجود دارد. اگر چنین نباشد، فریمهای تصدیق دریافت را هم نمی‌توان ارسال کرد. در پروتکل ۴ چنین فرضی وجود نداشت، و برای هر فریم یک فریم تصدیق دریافت مستقل پس فرستاده می‌شد. در پروتکل آینده این مشکل را هم بنحو جالبی حل خواهیم کرد.

دیدید که در پروتکل ۵، فرستنده می‌تواند در هر لحظه تعداد زیادی فریم ارسال شده (ولی هنوز تصدیق نشده) در بافر خود داشته باشد: هر یک از این فریمها یک تایمر مستقل می‌خواهند. این تایمرها را می‌توان بصورت نرم‌افزاری و با استفاده از وقفه‌های ساعت سخت‌افزاری، ایجاد کرد. این تایمرها تشکیل یک **لیست پیوندی** (linked list) می‌دهند، که هر گره آن سه بخش دارد: زمان باقی ماندهٔ تایمر، فریمی که به این تایمر مربوط است، و یک اشاره‌گر به گره بعدی.

در شکل ۳-۱۸ (الف) طرز پیاده‌سازی این تایمرها را می‌بینید. فرض کنید که ساعت سیستم هر 100 msec یک **تیک** (وقفهٔ سخت‌افزاری) می‌فرستد. در لحظهٔ شروع، که زمان واقعی 10:00:00.0 است، سه تایمر برای زمانهای 10:00:00.5 ، 10:00:01.3 و 10:00:01.9 ست می‌شوند. با هر تیک ساعت سخت‌افزاری، شمارندهٔ تایمری که در رأس لیست قرار دارد، کاهش می‌یابد. وقتی این شمارنده 0 شد، گره مربوطه از لیست حذف می‌شود؛ شکل ۳-۱۸(ب) را ببینید. سازماندهی تایمرها به صورت فوق باعث می‌شود که در هر بار فراخوانی روالهای *start_timer* و *stop_timer* کل لیست اسکن شود، ولی از سوی دیگر کار لازم برای به روز در آوردن تایمرها در هر تیک بسیار ناچیز است. همانطور که می‌بینید، در پروتکل ۵ روالهای *start_timer* و *stop_timer* پارامتری می‌گیرند، که نشان می‌دهد زمان کدام فریم بایستی سنجیده شود.



شکل ۳-۱۸. شبیه‌سازی چند تایمر بوسیلهٔ نرم‌افزار.

۳-٤-۳ پروتکل تکرار انتخابی

اگر نرخ خطا ناچیز باشد، پروتکل ۵ بخوبی کار خواهد کرد، اما در خطوط پُرنویز این پروتکل مقدار زیادی از پهنای باند را (برای ارسال مجدد فریمها) به هدر می‌دهد. استراتژی دیگر مقابله با خطاهای خط اینست که به گیرنده اجازه دهیم فریمهای سالمی که بعد از یک (یا چند) فریم معیوب از راه می‌رسند، را بافر کند. چنین پروتکلی دیگر فریمها را به صرف اینکه فریم قبلی آنها خراب بوده یا گم شده، دور نمی‌اندازد.

در این پروتکل، فرستنده و گیرنده هر دو پنجره‌ای از شماره‌های قابل قبول دارند. پنجرهٔ ارسال در فرستنده از 0 شروع شده، و می‌تواند تا *MAX_SEQ* بالا برود؛ اما اندازهٔ پنجرهٔ دریافت همیشه ثابت، و معادل *MAX_SEQ* است. گیرنده برای هر یک از فریمهایی که در پنجرهٔ دریافت قرار می‌گیرند، یک بافر ثابت دارد، که هر یک از این بافرها دارای بیتی هستند (بنام بیت *arrived*) که مشخص می‌کند بافر پُر است یا خالی. وقتی یک فریم بدست گیرنده می‌رسد، شمارهٔ آن را با تابع *between* چک می‌کند تا ببیند در داخل پنجرهٔ دریافت است یا خارج آن. اگر فریم داخل پنجره باشد و تکراری هم نباشد، پذیرفته شده و در بافر ذخیره می‌شود (خواه این فریم محتوی بسته‌ای باشد که باید در نوبت بعد به لایهٔ شبکه داده شود، یا نه). البته این فریم تا زمانی که تمام فریمهای قبل از آن به لایهٔ شبکه تحویل نشده‌اند، در لایهٔ پیوند داده می‌ماند. در شکل ۳-۱۹ پروتکلی که بر اساس این الگوریتم نوشته شده، را ملاحظه می‌کنید.

```
/* Protocol 6 (selective repeat) accepts frames out of order but passes packets to the
   network layer in order. Associated with each outstanding frame is a timer. When the
timer
   expires, only that frame is retransmitted, not all the outstanding frames, as in
protocol 5. */

   #define MAX_SEQ 7                                   /* should be 2^n - 1 */
   #define NR_BUFS ((MAX_SEQ + 1)/2)
   typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready, ack_timeout}
event_type;
   #include "protocol.h"
   boolean no_nak = true;                              /* no nak has been sent yet */
```

```
    seq_nr oldest_frame = MAX_SEQ + 1;          /* initial value is only for the simulator
*/
    static boolean between(seq_nr a, seq_nr b, seq_nr c)
    {
    /* Same as between in protocol5, but shorter and more obscure. */
      return ((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a));
    }
    static void send_frame(frame_kind fk, seq_nr frame_nr, seq_nr frame_expected,
packet buffer[])
    {
    /* Construct and send a data, ack, or nak frame. */
      frame s;                                  /* scratch variable */

      s.kind = fk;                              /* kind == data, ack, or nak */
      if (fk == data) s.info = buffer[frame_nr % NR_BUFS];
      s.seq = frame_nr;                         /* only meaningful for data frames */
      s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);
      if (fk == nak) no_nak = false;               /* one nak per frame, please */
      to_physical_layer(&s);                    /* transmit the frame */
      if (fk == data) start_timer(frame_nr % NR_BUFS);
      stop_ack_timer();                            /* no need for separate ack frame */
    }
    void protocol6(void)
    {
      seq_nr ack_expected;                      /* lower edge of sender's window */
      seq_nr next_frame_to_send;                   /* upper edge of sender's window +
1 */
      seq_nr frame_expected;                       /* lower edge of receiver's window */
      seq_nr too_far;                           /* upper edge of receiver's window + 1
*/
      int i;                                    /* index into buffer pool */
      frame r;                                  /* scratch variable */
      packet out_buf[NR_BUFS];                     /* buffers for the outbound stream */
      packet in_buf[NR_BUFS];                   /* buffers for the inbound stream */
      boolean arrived[NR_BUFS];                    /* inbound bit map */
      seq_nr nbuffered;                            /* how many output buffers currently
used */
      event_type event;

      enable_network_layer();                      /* initialize */
      ack_expected = 0;                            /* next ack expected on the inbound
```

```
stream */
     next_frame_to_send = 0;                    /* number of next outgoing frame */
     frame_expected = 0;
     too_far = NR_BUFS;
     nbuffered = 0;                             /* initially no packets are buffered */
     for (i = 0; i < NR_BUFS; i++) arrived[i] = false;
     while (true) {
        wait_for_event(&event);                 /* five possibilities: see event_type
above */
        switch(event) {
          case network_layer_ready:             /* accept, save, and transmit a new
frame */
               nbuffered = nbuffered + 1;       /* expand the window */
   from_network_layer(&out_buf[next_frame_to_send % NR_BUFS]); /* fetch
new packet */
   send_frame(data, next_frame_to_send, frame_expected, out_buf); /*
transmit the frame */
               inc(next_frame_to_send);         /* advance upper window edge */
               break;

          case frame_arrival:                   /* a data or control frame has arrived
*/
               from_physical_layer(&r);         /* fetch incoming frame from
physical layer */
               if (r.kind == data) {
                    /* An undamaged frame has arrived. */
                    if ((r.seq != frame_expected) && no_nak)
   send_frame(nak, 0, frame_expected, out_buf); else
start_ack_timer();
   if (between(frame_expected, r.seq, too_far) &&
(arrived[r.seq%NR_BUFS] == false)) {
                         /* Frames may be accepted in any order. */
                         arrived[r.seq % NR_BUFS] = true;  /* mark buffer as
full */
                         in_buf[r.seq % NR_BUFS] = r.info;  /* insert data into
buffer */
                         while (arrived[frame_expected % NR_BUFS]) {
                              /* Pass frames and advance window. */
                          to_network_layer(&in_buf[frame_expected % NR_BUFS]);
                              no_nak = true;
                              arrived[frame_expected % NR_BUFS] = false;
```

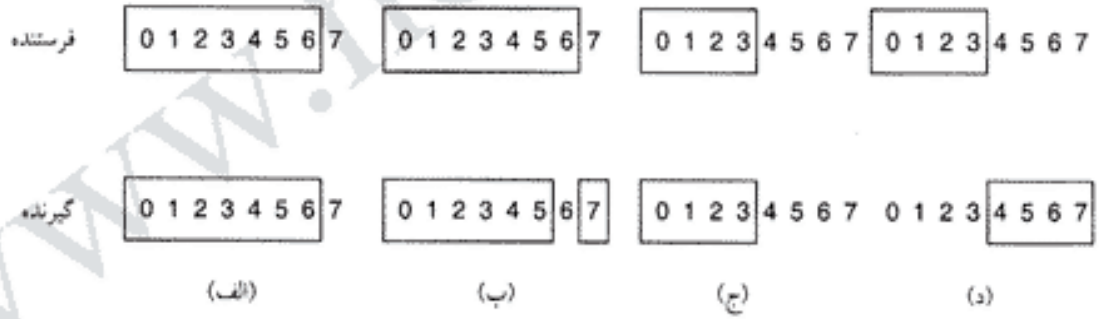
```
    inc(frame_expected);      /*      advance      lower      edge      of      receiver's
window */
                              inc(too_far);       /*  advance  upper  edge  of
receiver's window */
                              start_ack_timer();  /* to see if a separate ack is
needed */
                          }
                      }
                  }
    if((r.kind==nak)                                                          &&
 between(ack_expected,(r.ack+1)%(MAX_SEQ+1),next_frame_to_send))
    send_frame(data,     (r.ack+1)     %     (MAX_SEQ     +     1),     frame_expected,
out_buf);

                  while (between(ack_expected, r.ack, next_frame_to_send)) {
                      nbuffered = nbuffered - 1;    /* handle piggybacked ack */
                      stop_timer(ack_expected % NR_BUFS); /* frame arrived intact
*/
                      inc(ack_expected);            /* advance lower edge of sender's
window */
                  }
                  break;

          case cksum_err:
                  if (no_nak) send_frame(nak, 0, frame_expected, out_buf);/*  damaged
frame */
                  break;

          case timeout:
                  send_frame(data, oldest_frame, frame_expected, out_buf); /* we  timed
out */
                  break;

          case ack_timeout:
                  send_frame(ack,0,frame_expected, out_buf);      /* ack timer expired;
send ack */
          }
    ;)(reyal_krowten_elbasid esle ;)(reyal_krowten_elbane )SFUB_RN < dereffubn( fi
     }
   }
```

شکل ۳-۱۹. پروتکل پنجرهٔ لغزندهٔ «تکرار انتخابی».

دریافت نامنظم فریمها مسائلی را بوجود می‌آورد که در پروتکلهای قبلی (که فریمها را فقط بترتیب شماره قبول می‌کنند) وجود نداشت. با یک مثال بهتر می‌توان این مشکل را نشان داد. فرض کنید از شماره‌های ترتیبی سه‌بیتی استفاده می‌کنیم، بنابراین فرستنده قبل از توقف برای رسیدن اولین فریم تصدیق دریافت می‌تواند حداکثر هفت فریم ارسال کند. در لحظهٔ شروع، پنجره‌های ارسال و دریافت مانند شکل ۳-۲۰ (الف) هستند. فرستنده فریمهای 0 تا 6 را می‌فرستد. پنجرهٔ دریافت گیرنده فقط اجازهٔ پذیرش فریمهایی را می‌دهد که شمارهٔ آنها (منحصراً) بین 0 تا 6 باشد. تمام هفت فریم اول سالم به مقصد می‌رسند، بنابراین گیرنده دریافت آنها را تصدیق کرده، پنجرهٔ دریافت را برای دریافت سری بعدی فریمها (7، 0، 1، 2، 3، 4، یا 5) بجلو می‌برد، و تمام بافرها را هم با علامت «خالی» نشانه‌گذاری می‌کند؛ شکل ۳-۲۰ (ب) را ببینید.

درست در همین لحظه یک صاعقه به خط تلفن اصابت کرده، و تمام فریمهای تصدیق دریافت را (در راه بازگشت به فرستنده) از بین می‌برد. پس از مدتی تایمر فریم 0 به انتها رسیده، و فرستنده این فریم را مجدداً ارسال می‌کند. وقتی این فریم تکراری به گیرنده رسید، گیرنده چک می‌کند که آیا در داخل پنجرهٔ دریافت هست یا خیر. متأسفانه همانطور که در شکل ۳-۲۰ (ب) می‌بینید، فریم 0 هنوز در داخل پنجرهٔ دریافت قرار دارد، بنابراین گیرنده آنرا قبول می‌کند. از آنجائیکه فریمهای 0 تا 6 دریافت شده‌اند، گیرنده تصدیق دریافت فریم 6 را سوار فریم بعدی کرده و به فرستنده برمی‌گرداند.

فرستنده هم خوشحال از اینکه تمام فریمهای فرستاده شده سالم به مقصد رسیده‌اند، پنجرهٔ ارسال را بجلو برده و بلافاصله فریمهای 7، 0، 1، 2، 3، 4 و 5 را می‌فرستد. وقتی فریم 7 به مقصد رسید، لایهٔ پیوند داده گیرنده آنرا تحویل لایهٔ شبکه می‌دهد. بلافاصله پس از آن لایهٔ پیوند داده چک می‌کند که آیا بستهٔ 0 معتبری وجود دارد یا خیر، و چون چنین بسته‌ای در بافرهای آن موجود است، آنرا به لایهٔ شبکه می‌دهد؛ در حالیکه می‌دانیم این همان بستهٔ 0ول است و نباید دوباره به لایهٔ شبکه داده شود - پس پروتکل ما مرتکب خطا شده است.



شکل ۳-۲۰. (الف) پنجره‌های ارسال و دریافت هفت‌تایی در لحظهٔ شروع. (ب) بعد از رسیدن هفت فریم به مقصد، و قبل از بازگشت تصدیق دریافت به فرستنده. (ج) پنجره‌های ارسال و دریافت چهارتایی در لحظهٔ شروع. (د) بعد از رسیدن چهار فریم به مقصد، و قبل از بازگشت تصدیق دریافت به فرستنده.

منشأ این مشکل آنجاست که بعد از جلو رفتن پنجرهٔ دریافت در گیرنده، شماره‌های معتبر جدید با شماره‌های قدیمی **همپوشانی** (overlap) دارد. در نتیجه، فریمهای بعدی می‌توانند تکراری باشند (اگر تمام فریمهای تصدیق دریافت از بین برود) یا نباشند (اگر تمام فریمهای تصدیق دریافت سالم به فرستنده برسند). گیرندهٔ بیچاره هیچ راهی برای تشخیص این وضعیت ندارد.

راه چارهٔ این معضل آن است که مطمئن شویم بعد از جلو رفتن پنجرهٔ گیرنده، با پنجرهٔ اصلی همپوشانی نداشته باشد. برای رسیدن به این هدف پنجرهٔ دریافت باید از نصف تعداد شماره‌های ترتیبی تجاوز نکند؛ شکل

۳-۲۰ (پ) و (ت) را ببینید. برای مثال، اگر از شماره‌های ترتیبی ۴-بیتی استفاده کنیم، محدودهٔ ما 0 تا 15 خواهد بود، و فرستنده نباید در هر لحظه بیش از هشت فریم تصدیق نشده در بافر خود داشته باشد. بدین‌ترتیب، وقتی گیرنده فریمهای 0 تا 7 را گرفته و پنجرهٔ دریافت را جلو ببرد، به سری 8 تا 15 می‌رسد که آشکار است دیگر با مشکل قبلی مواجه نخواهد شد. در حالت کلی، اندازهٔ پنجره در پروتکل ۶ بایستی حداکثر $(MAX\_SEQ + 1)/2$ باشد. در مثال قبل، پنجرهٔ ارسال و دریافت را باید $(7 + 1)/2 = 4$ انتخاب کنیم.

و حالا یک سؤال جالب دیگر: گیرنده چند بافر باید داشته باشد؟ گیرنده در هیچ شرایطی فریمی که شمارهٔ آن کمتر از لبهٔ پائین پنجرهٔ دریافت یا بیشتر از لبهٔ بالای آن باشد، را قبول نخواهد کرد. در نتیجه، تعداد بافرهای گیرنده باید معادل اندازهٔ پنجرهٔ دریافت باشد (نه تعداد شماره‌های ترتیبی). با شماره‌های ترتیبی ۴-بیتی، گیرنده به حداکثر هشت بافر نیاز دارد. وقتی فریم i بدست گیرنده می‌رسد، آنرا در بافری بشمارهٔ i MOD 8 قرار می‌دهد. شاید حدس زده باشید که فریمهای i و $i + 8$ هر دو در یک بافر قرار می‌گیرند، ولی توجه کنید که این دو فریم هرگز در یک پنجره واقع نمی‌شوند (چون برای آنکه چنین اتفاقی بیفتد، اندازهٔ پنجره باید حداقل 9 باشد).

به دلیل مشابه، تعداد تایمرهای فرستنده نیز باید معادل پنجرهٔ دریافت باشد، نه تعداد شماره‌های ترتیبی (چون هر تایمر به یک بافر وابسته است و وقتی تایمر به انتها می‌رسد، بافر دوباره ارسال می‌شود).

در پروتکل ۵ این پیش‌فرض ضمنی را داشتیم که بار کانال بسیار زیاد است: وقتی یک فریم از راه می‌رسد، تصدیق آن بلافاصله برگردانده نمی‌شود تا سوار فریم بعدی (که از گیرنده به فرستنده می‌رود) شود. اما اگر ترافیک در جهت مخالف کم باشد، فریمهای تصدیق دریافت خیلی معطل خواهند شد. در این روش، وقتی فرستنده *MAX_SEQ* بسته فرستاد، منتظر می‌ماند و از آنجائیکه ترافیک جهت مقابل کم است، مدت زیادی بیکار خواهد ماند؛ علت فرض زیاد بودن بار کانال نیز همین موضوع است.

در پروتکل ۶ این مشکل نیز برطرف شده است. وقتی یک فریم (که طبق ترتیب مورد انتظار فرستاده شده) از راه رسید، گیرنده یک تایمر کمکی را (با تابع *start_ack_timer*) راه می‌اندازد. اگر در مدتی که این تایمر منقضی می‌شود، فریمی برای ارسال تحویل لایهٔ پیوند داده نشد، یک فریم تصدیق دریافت مستقل برگردانده خواهد شد (رویداد این تایمر *ack_timeout* نام دارد). با این تمهید دیگر نیازی نیست متکی به ترافیک سنگین دوطرفه باشیم، و پروتکل ۶ حتی می‌تواند بصورت کاملاً یکطرفه هم کار کند. از این تایمر کمکی فقط یکی وجود دارد، و اجرای تابع *start_ack_timer* آنرا ریست می‌کند. البته ضروری است که فاصلهٔ زمانی تایمر کمکی بسیار کوتاهتر از فاصلهٔ زمانی تایمرهای فریمهای داده باشد، تا این تایمرها قبل از رسیدن تصدیق دریافت منقضی نشوند.

استراتژی مقابله با خطا در پروتکل ۶ بسیار کارآمدتر از پروتکل ۵ است. اگر گیرنده به هر دلیلی ظن خطا ببرد، یک فریم **تصدیق دریافت منفی** (NAK) به فرستنده پس می‌فرستد. این NAK صریحاً از فرستنده می‌خواهد که فریم مشخص شده را دوباره ارسال کند. دو حالت وجود دارد که گیرنده باید به بروز خطا مشکوک شود: دریافت یک فریم معیوب، یا دریافت فریمی که انتظار آنرا ندارد. برای اجتناب از تکرار این درخواستها، گیرنده لیستی از فریمهایی که برای آنها NAK فرستاده را نگه می‌دارد. اگر متغیر *no_nak* در پروتکل ۶ مقدار true داشته باشد، نشان می‌دهد که هنوز برای فریم *frame_expected* هیچ NAK فرستاده نشده است. خراب یا گم شدن NAK ها مشکلی بوجود نخواهد آورد، فقط زمان ارسال مجدد فریمها را کمی به تأخیر می‌اندازد (چون تایمر فرستنده بهر حال منقضی خواهد شد). اگر پس از NAK فریم خواسته شده از راه برسد، گیرنده مقدار *no_nak* را به true ست کرده و تایمر کمکی را راه می‌اندازد (*start_ack_timer*). پس از انقضای این تایمر، گیرنده یک فریم ACK به فرستنده پس می‌فرستد، و بدین‌ترتیب خود را با آن سنکرون می‌کند.

در برخی شرایط، زمان لازم برای سفر فریمهای داده به مقصد، پردازش در آنجا، و بازگشت فریم تصدیق

دریافت (تقریباً) ثابت است. در چنین شرایطی فرستنده می‌تواند تایمرهای خود را کمی بالاتر از این مقدار تنظیم کند. اما اگر این زمان در حد وسیعی متغیر باشد، فرستنده دو راه در پیش رو دارد: فاصلهٔ زمانی تایمرهای خود را خیلی کوچک بگیرد (و ریسک ارسالهای تکراری را بپذیرد)، یا آنرا بسیار بزرگ بگیرد (و بعد از هر خطا مدت زیادی بیکار بماند). هر دوی این گزینه‌ها تلف کردن پهنای باند است.

اگر ترافیک جهت مخالف کم باشد، زمان برگشت فریمهای تصدیق دریافت نیز نامنظم خواهد بود (گاهی کم و گاهی زیاد). تغییر زمان پردازش فریم در گیرنده هم می‌تواند مزید علت باشد. در کل، اگر انحراف معیار فاصلهٔ زمانی فریم تصدیق دریافت (در مقایسه با کل فاصلهٔ زمانی) کوچک باشد، می‌توان فاصلهٔ زمانی تایمرها را «کوچک» در نظر گرفت، که در این حالت NAK سودمندی خود را از دست می‌دهد. در غیر اینصورت، برای اجتناب از تکرار در ارسال فریمها باید فاصلهٔ زمانی تایمرها را «بزرگ» گرفت؛ در این حالت استفاده از NAK می‌تواند ارسال مجدد فریمهای معیوب و گم شده را تسریع کند.

سؤال دیگری که در همین زمینه پیش می‌آید این است که: کدام فریم باعث انقضای تایمر شده است؟ پروتکل ۵ همیشه *ack_expected* است (یعنی فقط انتظار دریافت تصدیق را دارد)، چون همیشه قدیمی‌ترین باعث انقضای تایمر می‌شود. اما در پروتکل ۶ راه ساده‌ای برای تعیین این موضوع وجود ندارد. فرض کنید فریمهای 0 تا 4 فرستاده شده‌اند، و لیست فریمهای بافر شده در فرستنده از قدیم‌ترین به جدیدترین (از چپ براست) عبارتند از: 01234 . حال وضعیت زیر را در نظر بگیرید: فریم 0 منقضی می‌شود، فرستنده فریم (جدید) 5 را می‌فرستد، فریم *1* منقضی می‌شود، و فرستنده فریم (جدید) 6 را می‌فرستد. در این لحظه، بافر فرستنده حاوی فریمهای 3405126 (از قدیم به جدید) است. اگر در این لحظه تمام فریمهای برگشتی (که فرض می‌کنیم حامل فریمهای تصدیق دریافت هستند) از بین بروند، هفت فریمی که در بافر قرار دارند، به همین ترتیب منقضی می‌شوند.

برای اجتناب از پیچیدگی بیشتر (که تا همین جا هم باندازهٔ کافی پیچیده هست)، به مدیریت تایمرها نپرداختیم. بجای آن فرض کردیم که متغیر *oldest_frame* در لحظهٔ انقضای تایمر نشان می‌دهد که کدام فریم منقضی شده است.

۳-۵ ارزیابی پروتکل‌ها

پروتکلهای واقعی (و کُد نرم‌افزاری آنها) اغلب بسیار پیچیده‌اند، به همین دلیل تحقیقات زیادی انجام شده تا روشهای ریاضی و مشخصی برای ارزیابی آنها ابداع شود. در این قسمت برخی از این تکنیکها و مدلها را بررسی خواهیم کرد. با اینکه در اینجا برای ارزیابی پروتکلهای لایهٔ پیوند داده از این تکنیکها استفاده کرده‌ایم، ولی آنها را می‌توان برای لایه‌های دیگر نیز بکار گرفت.

۳-۵-۱ مدل ماشین حالت محدود

یکی از مفاهیم کلیدی در مدلسازی پروتکلها، **ماشین حالت محدود** (finite state machine) است. در این تکنیک، هر **ماشین پروتکل** (protocol machine) – یعنی، فرستنده یا گیرنده – در هر لحظه از زمان در حالتی خاص قرار دارد. این **حالت** (state) عبارتست از مقدار تمام متغیرها، و **شمارندهٔ برنامه** (program counter) .

در اکثر مواقع می‌توان تعداد زیادی از حالت‌ها را برای آنالیز دسته‌بندی کرد. برای مثال، گیرندهٔ پروتکل ۳ را در نظر بگیرید؛ تمام حالت‌های ممکن این گیرنده را می‌توان به دو دستهٔ مهم تقسیم کرد: انتظار برای فریم 0 ، و انتظار برای فریم 1 . تمام حالت‌های دیگر را می‌توان مراحل گذار از یکی از این دو حالت به حالت دیگر دانست. معمولاً حالت‌ها را بگونه‌ای انتخاب می‌کنند که در آن لحظه ماشین پروتکل در انتظار وقوع رویداد بعدی است (در مثال ما، اجرای روال *wait(event)*). در این لحظه، حالت پروتکل را می‌توان بطور کامل با دانستن مقدار متغیرهای آن

تعیین کرد. تعدادی حالت‌های چنین ماشینی 2^n است، که در آن n تعداد بیت‌های لازم برای نمایش تمام ترکیبات ممکنهٔ متغیرهای آن است.

حالت کل سیستم نیز عبارتست از ترکیب حالت‌های دو ماشین پروتکل (فرستنده و گیرنده) و حالت کانال. حالت کانال با محتویات آن تعیین می‌شود. اگر باز هم از پروتکل ۳ کمک بگیریم، کانال ما می‌تواند حالتهای زیر را داشته باشد: فریم 0 یا 1 از فرستنده به گیرنده می‌رود، فریم تصدیق دریافت از گیرنده به فرستنده برمی‌گردد، و یا کانال خالیست. اگر فرض کنیم گیرنده و فرستنده نیز هر کدام فقط دو حالت دارند، کل سیستم دارای ۱۶ حالت مجزا خواهد بود.

همین جا باید نکتهٔ کوچکی را دربارهٔ کانال توضیح دهیم. وقتی می‌گوئیم «فریم در کانال است»، البته از یک مفهوم مجرد حرف می‌زنیم. آنچه واقعاً منظور ماست، اینست که «فریم احتمالاً به مقصد رسیده، ولی هنوز پردازش نشده است». این فریم تا زمانی که ماشین پروتکل روال *FromPhysicalLayer* را اجرا نکرده و آنرا پردازش نکند، «در کانال می‌ماند».

هر حالت دارای تعدادی **گذار** (transition) به حالتهای دیگر است. گذار زمانی روی می‌دهد که رویدادی رخ دهد. در یک ماشین پروتکل، ارسال یک فریم، دریافت یک فریم، انقضای یک تایمر و مانند آن، همگی نمونه‌هایی از گذار هستند. رویدادهایی که می‌توانند در یک کانال رخ دهد، نیز عبارتند از: وارد شدن یک فریم جدید به کانال توسط ماشین پروتکل، برداشته شدن فریم از کانال توسط ماشین پروتکل، و یا گم شدن فریم در اثر نویز. با در دست داشتن توصیف کاملی از ماشینهای پروتکل و مشخصات کانال، می‌توان سیستم را بصورت نموداری از گره‌ها (حالت‌ها) و خطوط متصل‌کننده (گذارها) نمایش داد.

یکی از حالت‌های مهم در هر سیستم، **حالت اولیه** (initial state) است. این حالت متناظر است با وضعیت سیستم در لحظهٔ شروع به کار، یا (اگر مناسبتر باشد) کمی پس از آن. از این حالت اولیه می‌توان به کمک توالی گذارها به تمام (یا برخی از) حالت‌های دیگر رسید. با کمک تکنیکهای **نظریهٔ گراف** (graph theory) می‌توان مشخص کرد که کدام حالت‌ها قابل دسترسی‌اند، و کدامها نیستند. با این تکنیک، که به **آنالیز دسترسی** (reachability analysis) معروفست (Lin et al., 1987)، می‌توان دریافت که آیا یک پروتکل درست عمل می‌کند یا خیر.

مدل ماشین حالت محدود یک پروتکل را می‌توان بردار چهار عضوی (S, M, I, T) دانست، که در آن:

۱. S عبارتست از مجموعهٔ حالت‌هایی که پروسس‌ها و کانال می‌توانند در آن باشند.
۲. M عبارتست از مجموعهٔ فریمهایی که می‌توان روی کانال مبادله کرد.
۳. I عبارتست از مجموعهٔ حالت‌های اولیهٔ پروسس‌ها.
٤. T عبارتست از مجموعهٔ گذارهای بین حالت‌ها.

در لحظهٔ شروع، تمام پروسسها در حالت اولیه‌شان هستند. سپس رویدادها شروع به رخ دادن می‌کنند: فریمی برای ارسال آماده می‌شود، تایمرها خاموش می‌شوند، و مانند آن. هر رویداد می‌تواند باعث شود که یک پروسس یا کانال عملی را انجام داده و به حالت دیگر برود. با تعیین دقیق پیامدهای هر حالت، می‌توان گراف دسترسی را رسم و پروتکل را آنالیز کرد.

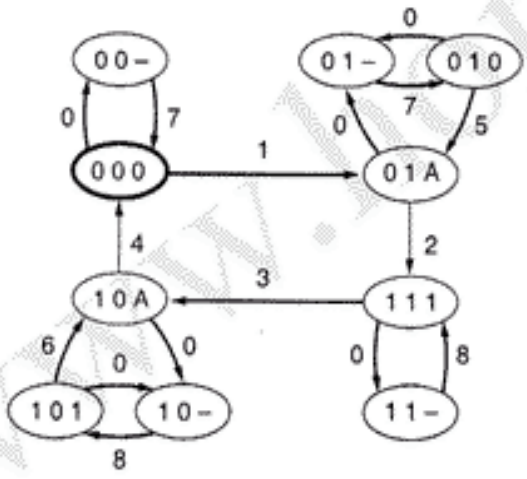
آنالیز دسترسی می‌تواند خطاهای مختلفی را در طراحی پروتکل روشن کند. برای مثال، اگر یک فریم خاص بتواند در حالت خاصی وجود داشته باشد و ماشین حالت محدود نتواند بگوید در این موقعیت چه باید کرد، طراحی ما مشکل دارد (ناقص است). اگر حالت یا حالتهایی وجود داشته باشد که نتوان از آن خارج شد (بعبارت دیگر، دریافت فریم سالم دیگر امکانپذیر نباشد)، باز هم پروتکل ما مشکل دارد (بن‌بست). موقعیت دیگری (که

در واقع مشکل چندان بزرگی نیست) آنست که پروتکل ما برای حالتهایی تجهیز شده باشد که امکان روی دادن آنها وجود ندارد (گذارهای نامربوط). گراف دسترسی خطاهای دیگر را هم می‌تواند کشف کند.

در شکل ۳-۲۱ (الف) نمونه‌ای از یک مدل ماشین حالت محدود را ملاحظه می‌کنید. این گراف معادل پروتکل ۳ است، که در بالا توضیح دادیم: هر ماشین پروتکل دارای دو حالت، و کانال دارای چهار حالت است. در کل ۱۶ حالت ممکن وجود دارد، که البته از حالت اولیه نمی‌توان به همهٔ آنها رسید. در شکل این حالتهای غیر قابل دسترسی را نشان نداده‌ایم، و برای سادگی کار از خطاهای **جمع تطبیقی** (checksum) هم چشم پوشیده‌ایم.

هر حالت با سه حرف *SRC* مشخص می‌شود، که در آن *S* یا 0 است یا 1 (متناظر با فریمی که فرستنده می‌خواهد بفرستد)؛ *R* نیز یا 0 است یا 1 (متناظر با فریمی که گیرنده منتظر دریافت آن است)؛ و *C* می‌تواند چهار مقدار 0 (فریم 0)، 1 (فریم 1)، A (فریم تصدیق دریافت) یا خالی (-) بگیرد (متناظر با حالتهای چهارگانهٔ کانال). در مثال بالا، حالت اولیه با (000) نشان داده شده است: یعنی فرستنده فریم 0 را فرستاده، گیرنده منتظر دریافت فریم 0 است، و فریم 0 اکنون در کانال است.

در شکل ۳-۲۱ نُه حالت گذار مختلف نشان داده شده است. در گذار 0 کانال محتویات خود را از دست می‌دهد. در گذار 1 کانال محتویات خود را به گیرنده تحویل می‌دهد، و گیرنده نیز حالت خود را به 1 (انتظار برای فریم 1) تغییر داده و یک فریم تصدیق دریافت به فرستنده پس می‌فرستد. گذار 1 همچنین متناظرست با تحویل بستهٔ 0 به لایهٔ شبکه در گیرنده. گذارهای دیگر را در شکل ۳-۲۱ (ب) ملاحظه می‌کنید. رسیدن فریمی با خطای جمع تطبیقی در اینجا نشان داه نشده، چون این اتفاق در پروتکل ۳ باعث تغییر حالت نمی‌شود.



(الف)

انتقال	نوبت کیست ؟	فریم پذیرفته شده	فریم ارسال شده	به لایه شبکه
0	–	(فریم گم شده)		–
1	R	0	A	Yes
2	S	A	1	–
3	R	1	A	Yes
4	S	A	0	–
5	R	0	A	No
6	R	1	A	No
7	S	(انقضای زمان)	0	–
8	S	(انقضای زمان)	1	–

(ب)

شکل ۳-۲۱. (الف) دیاگرام حالت پروتکل ۳. (ب) گذارها.

در حالت عادی، گذارهای 1، 2، 3 و 4 پشت سرهم و بارها و بارها تکرار می‌شوند. در هر سیکل دو بسته منتقل می‌شود، و با این کار فرستنده دوباره به حالت اولیه (ارسال فریم 0) برمی‌گردد. اگر فریم 0 در کانال از بین برود، سیستم از حالت (000) به حالت (-00) می‌رود (گذار 0). پس از مدتی تایمر فرستنده به انتها رسیده (گذار 7)، و سیستم به حالت (000) بازمی‌گردد. از بین رفتن فریم A (تصدیق دریافت) پیچیده‌تر است، و برای جبران آن به دو گذار نیاز داریم: 7 و 5، یا 6 و 8.

یکی از ویژگیهایی که یک پروتکل با شماره‌های ترتیبی 1-بیتی باید داشته باشد اینست که هرگز نباید دو فریم فرد متوالی (یا دو فریم زوج متوالی) به گیرنده برسد. در شکل ۳-۲۱ می‌توان این ویژگی را چنین نشان داد: «هیچ

مسیری از حالت اولیه وجود ندارد که طی آن دو گذار متوالی 1 رخ دهد، بدون اینکه بین آنها یک گذار 3 وجود داشته باشد، و بالعکس.» از شکل می‌توان دید که پروتکل ۳ این ویژگی را دارد.

الزام دیگر اینست که هیچ مسیری نباید وجود داشته باشد که طی آن فرستنده دو بار تغییر حالت دهد (از 0 به 1 و برگشت دوباره به 0) در حالیکه گیرنده ثابت مانده است. اگر چنین مسیری وجود داشته باشد، بمعنای آن است که دو فریم از بین رفته بدون اینکه گیرنده متوجه شده باشد.

ویژگی مهمتر یک پروتکل عدم وجود **بن‌بست** (deadlock) در آن است. بن‌بست حالتی است که پروتکل (تحت هیچ شرایطی) دیگر قادر به جلو رفتن نباشد (یعنی نتواند بسته‌ها را لایهٔ شبکه تحویل دهد). در مدل گراف، بن‌بست به زیرمجموعه‌ای از حالتها گفته می‌شود که بتوان از شرایط اولیه به آن رسید و دو ویژگی زیر را داشته باشد:

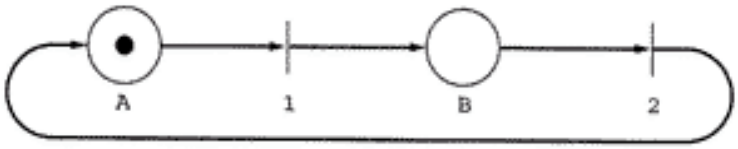
۱. **هیچ گذاری برای خروج از این زیرمجموعه وجود نداشته باشد.**
۲. **هیچ گذاری در داخل زیرمجموعه وجود نداشته باشد که باعث پیشرفت کار شود.**

وقتی یک پروتکل وارد بن‌بست شود، دیگر برای همیشه آنجا می‌ماند. باز هم از گراف شکل ۳-۲۱ می‌توان دید که (خوشبختانه) پروتکل ۳ هیچ بن‌بستی ندارد.

۳-۵-۲ مدل شبکهٔ پتری

ماشین حالت محدود تنها مدل برای ارزیابی پروتکلها نیست. در این قسمت به بررسی تکنیکی کاملاً متفاوت بنام **شبکهٔ پتری** (Petri net) می‌پردازیم (Danthine, 1980) . یک شبکهٔ پتری دارای چهار عنصر اساسی است: **مکان** (place) ، **گذار** (transition) ، **کمان** (arc) ، و **نشانه** (token) . مکان حالتیست که سیستم (یا بخشی از آن) می‌تواند در آن باشد. در شکل ۳-۲۲ یک شبکهٔ پتری را با دو مکان A و B می‌بینید، که با دایره مشخص شده‌اند. سیستم در حال حاضر در حالت A قرار دارد، که این موضوع با نشانه (نقطهٔ سیاه) در مکان A مشخص شده است. گذار با یک خط افقی یا عمودی مشخص می‌شود. هر گذار می‌تواند دارای تعدادی **کمان ورودی** (input arc – که از مکانهای ورودی آن می‌آیند) و تعدادی **کمان خروجی** (output arc – که به مکانهای خروجی آن می‌روند) باشد.

گذار **فعال** به گذاری گفته می‌شود که حداقل یک نشانه در یکی از ورودیهای آن وجود داشته باشد. گذار فعال می‌تواند هر گاه که اراده کند، **آتش** (fire) کرده و یک نشانه را از یکی از ورودیها برداشته و در یکی از خروجیهای خود قرار دهد. اگر تعداد کمانهای ورودی با کمانهای خروجی مساوی نباشد، نشانه **ابقا** (conserve) نخواهد شد.

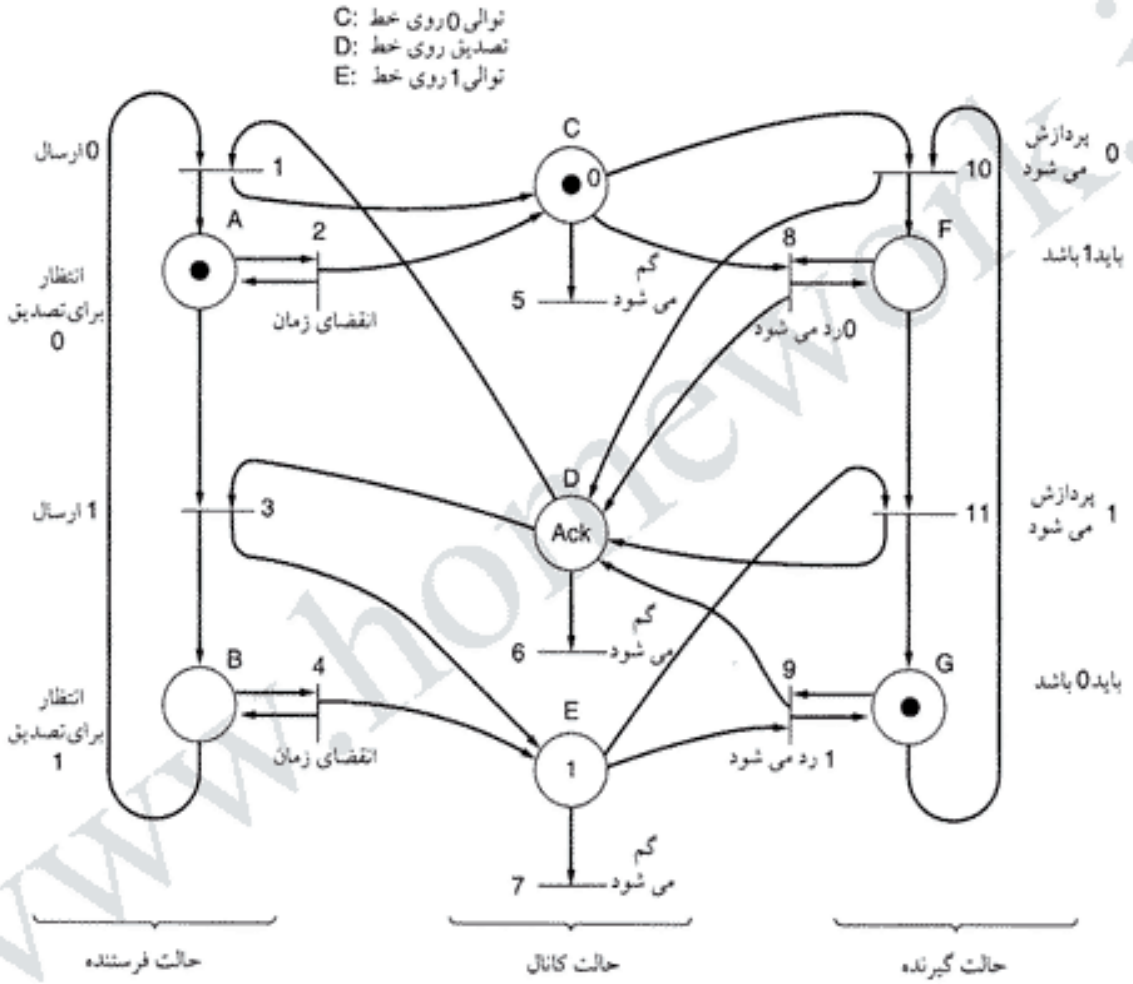


شکل ۳-۲۲. یک شبکهٔ پتری با دو مکان و دو گذار.

اگر دو یا چند گذار فعال وجود داشته باشد، هر کدام از آنها می‌توانند آتش کنند. اینکه کدام گذار آتش می‌کند نامشخص است، و همین ویژگیست که شبکهٔ پتری را برای مدلسازی پروتکل‌ها سودمند کرده است. شبکهٔ پتری شکل ۳-۲۲ کاملاً مشخص است و از آن فقط برای مدلسازی پروسسهایی که دو فاز بیشتر ندارند، می‌توان استفاده کرد (مانند رفتار یک نوزاد: خوردن، خوابیدن، خوردن، خوابیدن، و الی‌آخر). در اینجا هم مانند سایر تکنیکهای مدلسازی جزئیات زائد حذف می‌شوند.

در شکل ۳-۲۳ مدل شبکهٔ پتری شکل ۳-۱۲ (پروتکل ۳) را ملاحظه می‌کنید. بر خلاف مدل ماشین حالت

محدود، در اینجا حالتهای ترکیبی وجود ندارد؛ حالت فرستنده، گیرنده و کانال بطور مجزا و مستقل نمایش داده می‌شوند. گذارهای 1 و 2 بترتیب عبارتند از ارسال فریم 0 توسط فرستنده در حالت عادی، و بعد از انقضای تایمر. گذارهای 3 و 4 متناظر با ارسال فریم 1 در این دو موقعیت هستند. گذارهای 5 ، 6 و 7 نیز بترتیب از بین رفتن فریمهای 0 ، ACK و 1 را نشان می‌دهند. گذارهای 8 و 9 نشان می‌دهند که فریمی با شمارهٔ ترتیبی اشتباه (بترتیب 0 و 1) به گیرنده رسیده است. گذارهای 10 و 11 نیز بترتیب حاکی از رسیدن صحیح و سالم فریمهای 0 و 1 به گیرنده، و تحویل آنها به لایهٔ شبکه هستند.



شکل ۳-۲۳. مدل شبکهٔ پتری پروتکل ۳.

با شبکهٔ پتری نیز می‌توان مانند ماشین حالت محدود مشکلات یک پروتکل را تشخیص داد. برای مثال، اگر یک توالی آتش وجود داشته باشد که در آن دو گذار 10 (بدون یک گذار 11 بین آنها) رخ دهد، پروتکل ما مشکل دارد. مفهوم بن‌بست در شبکهٔ پتری نیز کاملاً شبیه همین مفهوم در ماشین حالت محدود است.

شبکهٔ پتری را بصورت جبری نیز می‌توان نوشت: هر گذار یک جملهٔ جبری است، که در یک دستور (که مکانهای ورودی و خروجی گذار را مشخص می‌کند) بکار می‌رود. از آنجائیکه شکل ۳-۲۳ دارای ۱۱ گذار است، برای نمایش جبری آن به ۱۱ دستور (که آنها را متناظر با گذارها شماره‌گذاری می‌کنیم) نیاز داریم. معادل جبری شبکهٔ پتری شکل ۳-۲۳ چنین است:

1: BD → AC
2: A → A
3: AD → BE
4: B → B
5: C →
6: D →
7: E →
8: CF → DF
9: EG → DG
10: CG → DF
11: EF → DG

همانطور که می‌بینید، یک پروتکل نسبتاً پیچیده به ۱۱ جملهٔ جبری ساده تبدیل شده که به آسانی می‌توان آنرا با کامپیوتر تحلیل کرد.

حالت فعلی شبکهٔ پتری با مجموعهٔ نامنظمی از مکانها (که هر مکان به تعداد نشانه‌هایی که دارد، ظاهر می‌شود) نشان داده می‌شود. در هر دستور، مکانهایی که سمت چپ جمله قرار دارند می‌توانند آتش کرده، و (بعد از حذف خود از حالت فعلی) خروجی خود را به حالت سیستم اضافه کنند. شکل ۳-۲۳ را می‌توان با علامت ACG نشان داد (یعنی، مکانهای A، C و G هر کدام یک نشانه دارند). در نتیجه، دستورات 2 ، 5 و 10 فعال هستند، و هر کدام از آنها می‌توانند اجرا شوند و حالت سیستم را عوض کنند (که البته حالت قبلی هم جزء حالتهای ممکنه است). توجه داشته باشید که، برای مثال، در این لحظه دستور 3 (AD → BE) نمی‌تواند اجرا شود، چون D در علامت سیستم وجود ندارد.

۳-۶ چند نمونه از پروتکلهای لینک داده

در این قسمت چند نمونه از پروتکلهای لینک داده را که کاربرد وسیعی دارند، مورد بررسی قرار خواهیم داد. اولین آنها، HDLC ، یک پروتکل **بیت-گرا** (bit-oriented) است که سالهاست در برنامه‌های بسیاری از ویرایشهای مختلف آن استفاده می‌شود. دومی، PPP ، یک پروتکل لینک داده است که برای اتصال کامپیوترهای خانگی به اینترنت بکار می‌رود.

۳-۶-۱ HDLC - کنترل سطح بالای لینک داده

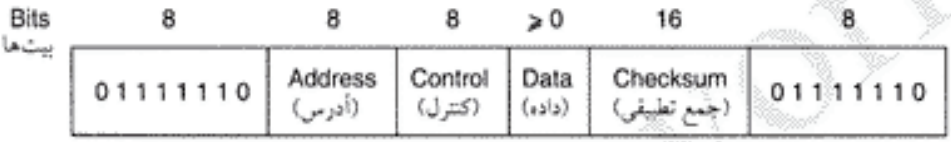
در این قسمت گروهی از پروتکلهای نزدیک به هم را بررسی می‌کنیم که با وجود قدیمی بودن، همچنان کاربرد گسترده‌ای دارند. همهٔ این پروتکلها از اولین پروتکل لینک داده که برای کامپیوترهای بزرگ IBM توسعه داده شد، مشتق شده‌اند: پروتکل SDLC (**کنترل لینک دادهٔ سنکرون** – Synchronous Data Link Control). بعد از توسعهٔ این پروتکل، IBM آنرا برای پذیرش بعنوان استاندارد آمریکایی و بین‌المللی به ANSI و ISO فرستاد. ANSI و ISO هر کدام تغییراتی در این پروتکل دادند، و بترتیب پروتکلهای ADCCP (**روش پیشرفتهٔ کنترل مخابرات داده** – Advanced Data Communication Control Procedure) و HDLC (**کنترل سطح بالای لینک داده** – High-Level Data Link Control) را از آن مشتق کردند. بعد از مدتی، CCITT تغییراتی در پروتکل HDLC داده، و پروتکل جدید را که LAP (**روش دسترسی لینک** – Link Access Procedure) نام گرفت، بعنوان قسمتی از استاندارد شبکه‌های X.25 معرفی کرد (این پروتکل بعدها برای سازگاری بهتر با ویرایشهای جدید HDLC باز هم اصلاح و LAPB نامیده شد). خوبی استانداردها همین تنوع زیاد آنهاست، و بالاخره می‌توانید یکی را مطابق سلیقه‌تان پیدا کنید؛ اگر هم پیدا نکردید، زیاد جای نگرانی نیست: سال آینده

مدلهای جدیدتری به بازار خواهد آمد!

تمام این پروتکلها مبنای واحدی دارند: همهٔ آنها بیت-گرا هستند، و از تکنیکهای **لاگذاری بیت** (bit stuffing) برای افزونگی داده‌ها استفاده می‌کنند. اختلاف آنها کوچک (ولی بهر حال، ناراحت‌کننده) است. برای اطلاع از مشخصات دقیق هر پروتکل می‌توانید به منابع مربوطه مراجعه کنید.

تمام پروتکلهای بیت-گرا از فریمهایی با ساختار شکل ۳-۲۴ استفاده می‌کنند. هر فریم با یک توالی پرچم (01111110) شروع می‌شود. فیلد **آدرس** (Address) در خطوطی اهمیت می‌یابد که ترمینالهای متعددی دارند، و از این فیلد برای مشخص کردن ترمینال مقصد استفاده می‌شود. در خطوط **نقطه-به-نقطه** (point-to-point) گاهی از این فیلد برای تشخیص **فرمان** (command) از **پاسخ** (response) استفاده می‌شود.

فیلد **کنترل** (Control) برای شمارهٔ ترتیبی فریم، تصدیق دریافت، و مقاصد دیگر بکار می‌رود (در ادامه این فیلد را بیشتر توضیح خواهیم داد).



شکل ۳-۲۴. فرمت فریم در پروتکلهای بیت-گرا.

فیلد **داده** (Data) محتوی اطلاعاتیست که فریم باید منتقل کند. طول این فیلد می‌تواند هر اندازه‌ای باشد، اگر چه با زیاد شدن آن کارایی تکنیک جمع تطبیقی (بدلیل بالا رفتن احتمال بروز خطاهای فورانی) کاهش خواهد یافت.

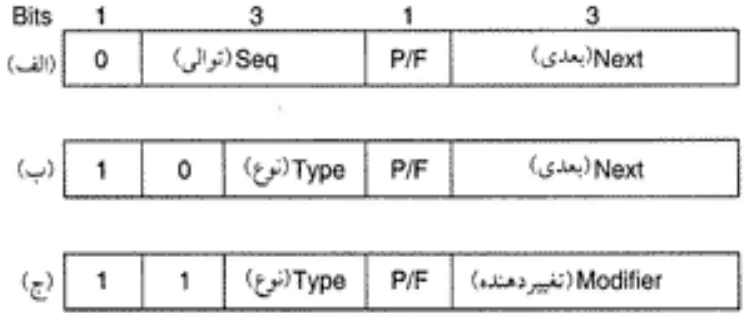
فیلد **جمع تطبیقی** (Checksum) یک **کُد افزونگی چرخه‌ای** (cyclic redundancy) است، که در قسمت ۳-۲-۲ توضیح دادیم.

در انتها، فریم به یک توالی پرچم دیگر (01111110) ختم می‌شود. وقتی یک خط نقطه-به-نقطه بیکار است، بطور پیوسته تواليهای پرچم را ارسال می‌کند. هر فریم باید حداقل سه فیلد (مجموعاً ۳۲ بیت) داشته باشد (البته منهای پرچمهای ابتدا و انتها).

فریمها بر سه نوعند: **اطلاعاتی** (Information)، **سرپرستی** (Supervisory)، و **بدون‌شماره** (Unnumbered). در شکل ۳-۲۵ فیلد کنترل هر یک از این سه نوع فریم را ملاحظه می‌کنید. این پروتکل از تکنیک پنجرهٔ لغزنده، با شماره‌های ترتیبی ۳-بیتی، استفاده می‌کند. در هر لحظه تا هفت فریم تصدیق نشده می‌تواند در بافر فرستنده وجود داشته باشد. فیلد *Seq* در شکل ۳-۲۵ (الف) شمارهٔ ترتیبی فریم را نشان می‌دهد. فیلد *Next* نیز فیلد سواری مجانی برای تصدیق دریافت است. با این حال، در تمام انوع پروتکلهای HDLC مرسوم است که بجای سوار کردن شمارهٔ آخرین فریم دریافت شده، شمارهٔ اولین فریمی که هنوز دریافت نشده (فریمی که گیرنده منتظر آن است) برگردانده شود. این دو روش هیچ مزیتی بر یکدیگر ندارند، و انتخاب یکی از آنها به طراح پروتکل بستگی دارد؛ البته مشروط باینکه همواره از یک روش استفاده کند.

فیلد *P/F* مخفف *Poll/Final* (سرکشی/پایان) است. این فیلد در کامپیوترها یا مودمهایی بکار می‌رود که به چندین ترمینال **سرکشی** (polling) می‌کنند. اگر این فیلد محتوی *P* باشد، کامپیوتر (یا مودم) ترمینال را دعوت به ارسال داده می‌کند. در تمام فریمهایی که ترمینال می‌فرستد، فیلد *P/F* مقدار *P* دارد (بجز در آخرین فریم، که مقدار آن *F* است).

در برخی از پروتکلها بیت *P/F* باعث می‌شود تا ماشین طرف مقابل بلافاصله فریم سرپرستی را بفرستد، و منتظر سواری مجانی نشود. در ارتباطاتی که از فریمهای بدون‌شماره سود می‌برند، نیز این بیت کاربرد دارد.



شکل ۳-۲۵. فیلد کنترل در یک (الف) فریم اطلاعاتی، (ب) فریم سرپرستی، (ج) فریم بدون‌شماره.

انواع مختلف فریمهای سرپرستی با فیلد *Type* مشخص می‌شود. نوع 0 فریم تصدیق دریافت (که رسماً RECEIVE READY نامیده می‌شرد) است، و مشخص می‌کند که گیرنده آمادهٔ دریافت فریم بعدیست. این فریم وقتی ارسال می‌شود که ترافیک جهت برگشت برای اجرای تکنیک سواری مجانی وجود نداشته باشد.

نوع 1 فریم تصدیق دریافت منفی (که رسماً REJECT نامیده می‌شود) است، و از آن برای تصدیق دریافت فریم با خطا استفاده می‌شود. در این حالت فیلد *Next* حاوی اولین فریمیست که درست دریافت نشده است (یعنی فریمی که باید دوباره ارسال شود). در اینجا فرستنده باید تمام فریمهای بعد از فریم *Next* را مجدداً بفرستد، و از این نظر شبیه پروتکل 5 است تا پروتکل 6.

نوع 2 فریم RECEIVE NOT READY است؛ این فریم اعلام می‌کند که تمام فریمهای قبل از *Next* درست دریافت شده‌اند، ولی خود *Next* خیر (و از این نظر شبیه RECEIVE READY است، با این تفاوت که جلوی ادامهٔ ارسال را می‌گیرد). فریم RECEIVE NOT READY در واقع نوعی اعلام مشکل است از سوی گیرنده، مثلاً مشکل پُر شدن بافرها. بعد از برطرف شدن این وضعیت، گیرنده یکی از فریمهای کنترلی دیگر (مانند RECEIVE READY یا REJECT) را می‌فرستد.

نوع 3 فریم SELECTIVE REJECT است. این فریم فقط زمانی فرستاده می‌شود که گیرنده خواستار ارسال مجدد یک فریم خاص باشد؛ HDLC از این نظر شبیه پروتکل 6 است تا پروتکل 5، و برای مواقعی مفید است که اندازهٔ پنجرهٔ ارسال نصف شمارهٔ ترتیبی باشد. بنابراین اگر گیرنده بخواهد فریمهای نامنظم را بافر کرده و فقط ارسال مجدد برخی از فریمهای معیوب را طلب کند، می‌تواند از SELECTIVE REJECT استفاده کند. این فریم کنترلی فقط در HDLC و ADCCP وجود دارد، ولی در SDLC و LAPB تعریف نشده است.

سومین نوع از فریمها، فریمهای **بدون‌شماره** (Unnumbered) است. این نوع گاهی کاربردهای کنترلی دارد، ولی در سرویسهای غیرقابل‌اعتماد غیرمتصل نیز می‌توان از آن برای برای انتقال داده استفاده کرد. برخلاف دو نوع دیگر، این نوع فریم عملکردهای متفاوتی در انواع پروتکلهای بیت-گرا است. در این حالت پنج بیت برای تعیین کارکرد فریم وجود دارد، ولی تمام ۳۲ حالت آن استفاده نمی‌شود.

در تمام پروتکلها فرمانی وجود دارد بنام DISC (**قطع ارتباط**) که اجازه می‌دهد یک ماشین خاموش شدن خود را به ماشینهای دیگر اعلام کند. فرمان دیگری وجود دارد که به یک ماشین اجازه می‌دهد تا بازگشت خود را اعلام کرده، و تمام شماره‌های ترتیبی را به 0 برگرداند؛ این فرمان SNRM (**ست شدن حالت پاسخ عادی** – Set Normal Response Mode) نام دارد. متأسفانه، «حالت پاسخ عادی» هر چیزی هست جز عادی. این حالت عبارتست از یک رابطهٔ نامتقارن، که در آن یک سر خط «ارباب» و سر دیگر خط «رعیت» است. این فرمان از زمانی

به جای مانده که یک کامپیوتر بزرگ مرکزی وجود داشت و تعداد زیادی ترمینال به آن متصل می‌شد، که در این حالت مسلماً رابطه نامتقارن ارباب و رعیتی «عادی» محسوب می‌شود. برای آن که این پروتکلها بتوانند پاسخگوی نیازهای جدید (رابطهٔ متقارن) باشند، در HDLC و LAPB فرمان دیگری وجود دارد بنام SABM (**ست شدن حالت آسنکرون متعادل** – Set Asynchronous Balanced Mode)، که خط را ریست کرده و دو سر آنرا به حالت هم‌ارز و متعادل درمی‌آورد. در این پروتکلها دو فرمان دیگر وجود دارد بنامهای SABME و SNRME که بترتیب با SABM و SNRM متناظر هستند، و فقط در آنها شمارهٔ ترتیبی فریمها (بجای ۳-بیتی) ۷-بیتی است.

فرمان سومی که در تمام پروتکلها وجود دارد، FRMR (FRaMe Reject) است که نشان می‌دهد جمع تطبیقی فریم صحیح است ولی از نظر شکلی درست نیست. از میان چنین شکلهای نادرستی می‌توان به فریم سرپرستی نوع 3 در LAPB، فریمی که کوتاهتر از ۳۲ بیت است، و تصدیق دریافت فریمی که خارج از پنجرهٔ دریافت است، اشاره کرد. در فریم FRMR یک فیلد دادهٔ ۲۴ بیتی وجود دارد که علت خطا را توضیح می‌دهد. اطلاعاتی که در این فیلد مخابره می‌شود، عبارتند از: فیلد کنترل فریم معیوب، پارامترهای پنجرهٔ دریافت، و اطلاعاتی دربارهٔ طبیعت خطا.

احتمال خراب یا گم شدن فریمهای تصدیق دریافت نیز مانند سایر فریمها وجود دارد، بنابراین برای آنها نیز باید نوعی تصدیق دریافت پس فرستاد. برای این منظور فریم کنترلی خاصی بنام UA (**تصدیق دریافت بدون‌شماره** – Unnumbered Acknowledgement) در نظر گرفته شده است. از آنجائیکه همیشه فقط یک فریم تصدیق دریافت تصدیق نشده در گیرنده وجود دارد، هیچ ابهامی وجود ندارد که منظور از یک فریم UA کدام فریم تصدیق دریافت است.

سایر فریمهای کنترلی کارهایی مانند **آماده‌سازی** (initialization)، **سرکشی** (polling) و **گزارش وضعیت** (status report) انجام می‌دهند. فریم کنترلی دیگری نیز وجود دارد بنام UI (Unnumbered Information)، که می‌توان با آن اطلاعات دلخواه ارسال کرد؛ این اطلاعات برای لایهٔ پیوند داده هستند، و به لایهٔ شبکه داده نمی‌شوند.

پروتکل HDLC علیرغم کاربرد وسیع آن، بهیچوجه کامل نیست. در (Fiorini et al., 1994) می‌توانید بحثی دربارهٔ مشکلات این پروتکل را ببینید.

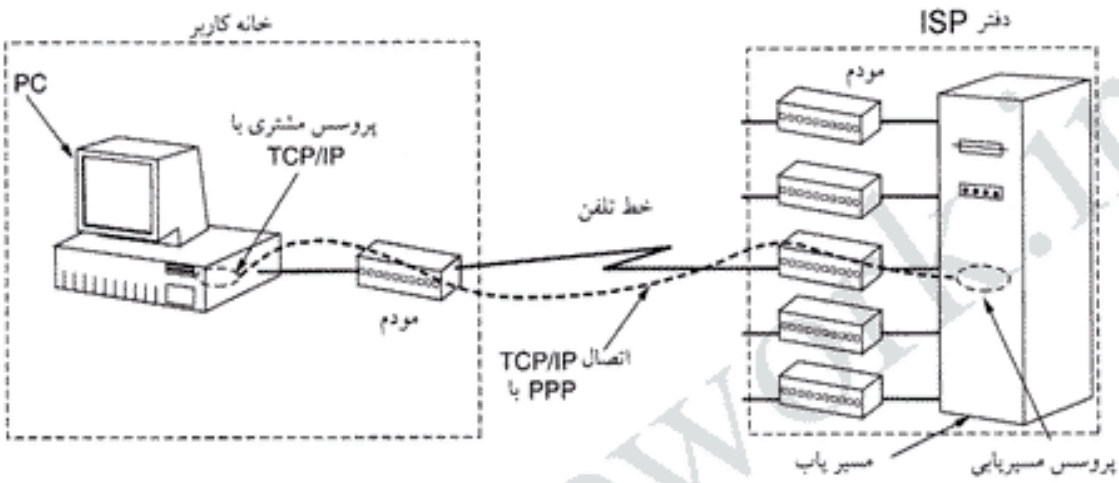
۳-۶-۲ لایهٔ پیوند داده در اینترنت

اینترنت شامل ماشینهای متعددی (میزبان و مسیریاب) است، که توسط یک ستون فقرات به یکدیگر متصل می‌شوند. در یک ساختمان کوچک می‌توان از تکنیکهای LAN برای ارتباط استفاده کرد، ولی در اینترنت اغلب ارتباطات از نوع **نقطه-به-نقطه** (piont-to-point) است. در فصل ۴ دربارهٔ لایهٔ پیوند داده در LAN صحبت خواهیم کرد؛ در این قسمت به لایهٔ پیوند داده در خطوط نقطه-به-نقطه می‌پردازیم.

در عمل، ارتباط نقطه-به-نقطه در دو حالت بکار برده می‌شود. اول، هزاران شرکت و مؤسسه دارای شبکه‌های محلی با تعداد زیادی ماشین میزبان (کامپیوترهای رومیزی، ایستگاههای کاری، کامپیوترهای سرویس‌دهنده و غیره) و یک مسیریاب (یا یک **پل** – bridge – که در عمل همان وظیفه را انجام می‌دهد) هستند، و این مسیریابها در یک شبکهٔ بزرگتر بیکدیگر متصل شده‌اند. معمولاً این مسیریابها بوسیلهٔ ارتباط نقطه-به-نقطه و از طریق **خطوط اجاره‌ای** (leased line) به یک (یا دو) مسیریاب دیگر متصل می‌شوند. همین مسیریابها و خطوط اجاره‌ای هستند که زیرشبکهٔ اینترنت را می‌سازند.

دومین حالتی که ارتباط نقطه-به-نقطه بکار برده می‌شود، میلیونها کاربر اینترنتی هستند که از منزل و با یک مودم (از طریق خط تلفن) به اینترنت متصل می‌شوند. اتفاقی که معمولاً می‌افتد اینست که کامپیوتر کاربر به

مسیریاب سرویس‌دهندهٔ اینترنت (ISP) زنگ می‌زند، و از آن طریق (درست مثل یک میزبان معمولی) به اینترنت متصل می‌شود. در این روش فرقی نمی‌کند که خط تلفن معمولی است یا اجاره‌ای، فقط بعد از اینکه کاربر دیگر نیازی به آن نداشت، ارتباط قطع می‌شود. در شکل ۳-۲۶ این نوع ارتباط نقطه-به-نقطه را ملاحظه می‌کنید. مودمی که در این شکل نشان داده‌ایم یک مودم خارجی است، اما مودمهای داخلی نیز دقیقاً همان کار را انجام می‌دهند.



شکل ۳-۲۶. یک کامپیوتر خانگی می‌تواند نقش میزبان اینترنت را بازی کند.

در هر دو حالت (ارتباط مسیریاب-مسیریاب یا میزبان-مسیریاب) به یک پروتکل لینک دادهٔ نقطه-به-نقطه نیاز داریم تا وظایفی از قبیل فریم‌بندی، کنترل خطا و مانند آن را انجام دهد. پروتکلی که در اینترنت از آن استفاده می‌شود (و در این قسمت آنرا بررسی خواهیم کرد)، PPP نام دارد.

PPP - پروتکل نقطه-به-نقطه

اینترنت در موارد مختلفی، از قبیل ترافیک مسیریاب-به-مسیریاب یا ترافیک کاربر-به-ISP، به پروتکل نقطه-به-نقطه نیاز دارد. این پروتکل PPP (**پروتکل نقطه-به-نقطه** - Point-to-Point Protocol) نام دارد، که در RFC 1661 تعریف شده و در چند RFC دیگر (از قبیل RFC 1662 و RFC 1663) مشخصات آن بهبود یافته است. PPP ویژگیهای کنترل خطا دارد، از پروتکلهای مختلف پشتیبانی می‌کند، اجازه می‌دهد تا آدرس IP در زمان اتصال به طرف مقابل درخواست شود، **تعیین هویت** (authentication) انجام می‌دهد، و دهها ویژگی دیگر. مهمترین بخشهای PPP عبارتند از:

۱. یک روش فریم‌بندی، که ابتدا و انتهای فریمها را بوضوح مشخص می‌کند. فرمت فریم در PPP تشخیص خطا را نیز انجام می‌دهد.
۲. یک پروتکل کنترل لینک برای برقراری ارتباط، تست آن، مذاکره برای سایر گزینه‌ها، و در پایان قطع ارتباط بصورتی آبرومندانه. این پروتکل LCP (پروتکل کنترل لینک - Link Control Protocol) نام دارد. LCP از هر دو ارتباط سنکرون و آسنکرون، و بیت-گرا یا بایت-گرا پشتیبانی می‌کند.
۳. روشی برای ارتباط و مذاکره دربارهٔ گزینه‌های لایهٔ شبکه، که از طرز کار این لایه مستقل است. در این روش برای هر نوع لایهٔ شبکه یک NCP (پروتکل کنترل شبکه - Network Control Protocol) وجود دارد.

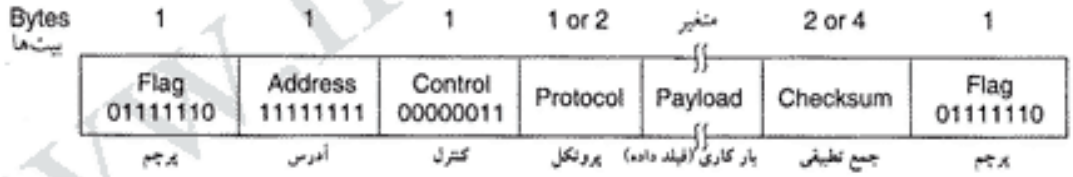
برای اینکه ببینید این قطعات چگونه با یکدیگر جفت می‌شوند، سناریوی ارتباط کاربر-به-ISP را در نظر می‌گیریم. ابتدا PC کاربر از طریق مودم خود به مسیریاب ISP زنگ می‌زند. بعد از اینکه مودم مسیریاب گوشی را

برداشت و ارتباط برقرار شد، PC از طریق فیلد دادهٔ یک یا چند فریم PPP چند بستهٔ LCP به مسیریاب می‌فرستد. پارامترهای PPP از طریق همین بسته‌ها و پاسخ آنها انتخاب می‌شود.

بعد از آنکه بر سر این پارامترها توافق حاصل شد، یک سری بستهٔ NCP برای پیکربندی لایهٔ شبکه رد و بدل می‌شود. معمولاً PC هایی که به اینترنت متصل می‌شوند از TCP/IP استفاده می‌کنند، پس PC ما به یک آدرس IP نیاز دارد. تعداد آدرسهای IP آنقدر زیاد نیست که بتوان به همه آدرس ثابت اختصاص داد، بهمین دلیل ISP تعدادی آدرس IP را بصورت دینامیک به PC هایی که به آن وصل می‌شوند، اختصاص می‌دهد. اگر یک ISP دارای n آدرس IP باشد، می‌تواند در هر لحظه (حداکثر) n کاربر متصل به اینترنت داشته باشد (البته تعداد کل مشترکان آن می‌تواند خیلی بیشتر باشد). یکی از بسته‌های NCP که درخواست IP می‌کند، این آدرس را به PC اختصاص می‌دهد.

از این لحظه به بعد PC ما درست مثل یک میزبان معمولی اینترنت است، و می‌تواند بسته‌های IP رد و بدل کند. وقتی کاربر ارتباط را قطع کند، NCP نیز ارتباط لایهٔ شبکه را قطع کرده و آدرس IP را آزاد می‌کند. پس از آن LCP ارتباط لایهٔ پیوند داده را قطع می‌کند، و کامپیوتر نیز به مودم دستور می‌دهد که گوشی را بگذارد و به ارتباط فیزیکی پایان دهد.

فرمت فریمهای PPP بسیار شبیه HDLC است (همیشه که نباید چرخ را از نو اختراع کرد). تفاوت اصلی PPP و HDLC اینست که، PPP **کاراکتر-گرا** (character-oriented) است نه بیت-گرا. بویژه، PPP از تکنیک **لاگذاری بایت** (byte stuffing) استفاده می‌کند، بنابراین تعداد بایتهای یک فریم همیشه عددی صحیح است. در PPP (برخلاف HDLC) نمی‌توان فریمی فرستاد که مثلاً 30.25 بایت داشته باشد. البته PPP می‌تواند (علاوه بر خطوط تلفن معمولی) روی SONET یا خطوط بیت-گرای HDLC نیز کار کند. فرمت فریم PPP را در شکل ۳-۲۷ ملاحظه می‌کنید.



شکل ۳-۲۷. فرمت فریم کامل PPP برای حالت بدون‌شماره.

تمام فریمهای PPP با بایت پرچم استاندارد HDLC (یعنی 01111110)، شروع می‌شوند، که اگر این بایت در داخل داده‌ها وجود داشته باشد از تکنیک لاگذاری بایت برای متمایز کردن آن استفاده می‌شود. بعد از آن فیلد *Address* می‌آید، که همیشه 11111111 است و مشخص می‌کند که تمام گیرنده‌ها باید این فریم را قبول کنند. با استفاده از این آدرس مشکل تخصیص آدرس‌های لینک داده نیز هم حل می‌شود.

بدنبال آدرس فیلد *Control* قرار دارد، که مقدار پیش‌فرض آن 00000011 است، و نشان می‌دهد که فریمها **بدون‌شماره** (unnumbered) هستند. بعبارت دیگر، در PPP فریمها شمارهٔ ترتیبی ندارند، و از فریم تصدیق دریافت نیز خبری نیست. البته در محیط‌های پُرنویز (مانند لینکهای بیسیم) می‌توان از فریمهای شماره‌دار استفاده کرد. جزئیات دقیق این حالت در استاندارد RFC 1663 مشخص شده، ولی در عمل بندرت از آن استفاده می‌شود.

از آنجائیکه در پیکربندی پیش‌فرض فیلدهای *Address* و *Control* همواره ثابت هستند، LCP مکانیزم خاصی را بین دو سر خط پیاده می‌کند که این دو بایت بکلی حذف و فریمها کوتاهتر شوند.

فیلد چهارم PPP فیلد *Protocol* است، و مشخص می‌کند که دادهٔ موجود در فریم (قسمت Payload) از چه

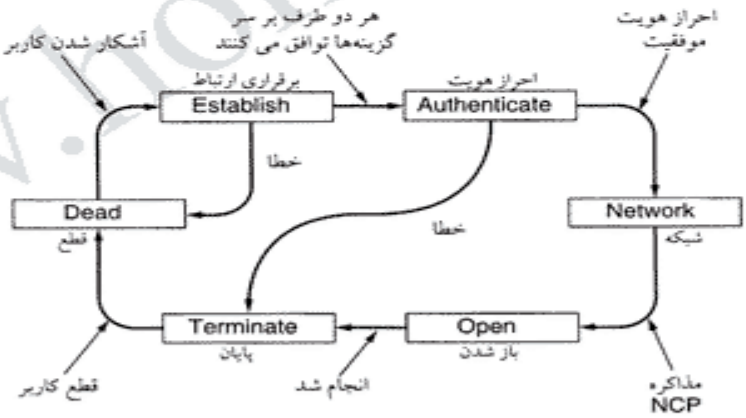
نوعیست. برای پروتکلهای مختلف از قبیل LCP ، NCP ، IP ، IPX ، Apple Talk و پروتکلهای دیگر تُدهایی تعریف شده است. پروتکلهایی که با 0 شروع می‌شوند، پروتکلهای لایهٔ شبکه (مانند IP ، IPX ، OSI CLNP و XNS) هستند؛ آنهایی که با 1 شروع می‌شوند، برای مذاکره در بارهٔ پروتکلهای دیگر بکار می‌روند (از جمله LCP ، و یک NCP خاص برای هر یک از انواع لایه‌های شبکه). اندازهٔ پیش‌فرض این فیلد 2 بایت است، ولی دو طرف می‌توانند از طریق LCP مذاکره کرده و اندازهٔ آنرا به 1 بایت تقلیل دهند.

طول فیلد *Payload* متغیر است، و حداکثر مقدار آن در مذاکرهٔ اولیه مشخص می‌شود. اگر دو طرف در مذاکرهٔ اولیه (هنگام برقراری ارتباط) بر سر این عدد به توافق نرسند، از عدد پیش‌فرض 1500 بایت استفاده خواهد شد. اگر مقدار دادهٔ ارسالی به این حد نرسد، لایهٔ پیوند داده بقیه را با کاراکتر خاصی پُر خواهد کرد.

بدنبال *Payload* فیلد *Checksum* می‌آید، که مقدار آن معمولاً 2 بایت است، ولی دو طرف می‌توانند بر سر جمع تطبیقی 2 بایتی هم توافق کنند.

بطور خلاصه، PPP یک مکانیزم فریم‌بندی چندپروتکلی است، که می‌توان از آن روی مودم، خطوط بیت-گرای HDLC ، SONET و سایر لایه‌های فیزیکی استفاده کرد. این پروتکل از کشف خطا، مذاکره برای گزینه‌های مطلوب، **فشرده‌سازی سرآیند** (header compression) (و در صورت نیاز، از ارتباط قابل‌اعتماد یا فرمت HDLC) پشتیبانی می‌کند.

اکنون اجازه دهید ببینیم در PPP برقراری خط و قطع ارتباط چگونه انجام می‌شود. در شکل ۳-۲۸ مراحل (ساده شدهٔ) برقراری خط، بکارگیری، و قطع آن نشان داده شده است. این مراحل برای ارتباط مودمی و یا مسیریاب-به-مسیریاب هر دو صادق است.



شکل ۳-۲۸. مراحل ساده شدهٔ برقرای و قطع خط در پروتکل PPP .

در شروع کار پروتکل خط در وضعیت *DEAD* است، و این به معنای آنست که هیچگونه کاربر یا ارتباطی در لایهٔ فیزیکی وجود ندارد. بعد از برقرای ارتباط در لایهٔ فیزیکی، خط به وضعیت *ESTABLISH* می‌رود. در این‌لحظه مذاکره بر سر گزینه‌های LCP شروع می‌شود، که اگر موفقیت‌آمیز باشد، به وضعیت *AUTHENTICATE* منجر می‌شود. حال دو طرف می‌توانند در صورت تمایل هویت یکدیگر را چک کنند. بعد از ورود به مرحلهٔ *NETWORK* ، لایهٔ شبکه با اجرای پروتکل NCP مناسب پیکربندی می‌شود. اگر این پیکربندی موفقیت‌آمیز

باشد، مرحلهٔ *OPEN* فرا رسیده و تبادل اطلاعات می‌تواند شروع شود. وقتی تبادل اطلاعات انجام شد، خط به مرحلهٔ *TERMINATE* رفته، و از آنجا دوباره به وضعیت *DEAD* برمی‌گردد و کاربر قطع می‌شود.

در مرحلهٔ *ESTABLISH*، مذاکره بر سر گزینه‌های پروتکل لینک داده توسط LCP انجام می‌شود. البته خود LCP هیچ علاقه‌ای به این گزینه‌ها ندارد، و فقط مکانیزمی برای مذاکره فراهم می‌آورد (بعبارت دیگر، پیشنهادی را فرستاده و بعد از دریافت پاسخ آنرا پذیرفته یا رد می‌کند). بررسی کیفیت خط (و اینکه آیا برای برقراری ارتباط باندازهٔ کافی خوب هست یا نه) نیز بر عهدهٔ LCP است. قطع خط (بعد از پایان یافتن کار) یکی دیگر از وظایف پروتکل LCP است.

در RFC 1661 یازده نوع فریم LCP تعریف شده است، که آنها را در شکل ۳-۲۹ ملاحظه می‌کنید. چهار فریم *Configure-* به **آغازکننده** (I) اجازه می‌دهند تا گزینه‌ای را پیشنهاد کند، و **پاسخ‌دهنده** (R) می‌تواند آنها را پذیرفته یا رد کند. اگر پاسخ‌دهنده گزینه‌ای را رد کند، می‌تواند پیشنهاد موردنظر خود را ارائه کرده، و یا اعلام کند که اساساً مایل نیست راجع به آن مذاکره کند. گزینه‌ها و پاسخ آنها جزئی از فریمهای LCP هستند.

نام	جهت	توضیح
Configure-request	I → R	لیست گزینه ها و مقادیر پیشنهادی
Configure-ack	I ← R	تمام گزینه ها قبول می شوند
Configure-nak	I ← R	بعضی از گزینه ها قبول نمی شوند
Configure-reject	I ← R	بعضی از گزینه ها مذاکره نمی شوند
Terminate-request	I → R	تقاضای قطع خط
Terminate-ack	I ← R	قبول، خط قطع شد
Code-reject	I ← R	دریافت درخواست نامعلوم
Protocol-reject	I ← R	درخواست پروتکل نامعلوم
Echo-request	I → R	لطفا این فریم را پس بفرست
Echo-reply	I ← R	فریم پس فرستاده شد
Discard-request	I → R	این فریم را نادیده بگیر (برای تست بود)

شکل ۳-۲۹. انواع فریمهای LCP .

کُدهای *Terminate-* برای قطع کردن خط (وقتی که دیگر نیازی به آن نیست) هستند. کُدهای *Code-reject* و *Protocol-reject* مشخص می‌کنند که پاسخ‌دهنده چیزی را دریافت کرده که معنی آنرا نمی‌فهمد. یکی از دلایل این وضعیت می‌تواند رخ دادن خطاهای کشف نشده روی خط باشد، ولی باحتمال بیشتر دلیل آن یکی نبودن ویرایش پروتکل LCP در دو سمت خط است. فریمهای *Echo-* برای تست کیفیت خط بکار برده می‌شوند. و بالاخره، فریم *Discard-request* برای دیباگ کردن بکار می‌آید (و نویسندهٔ پروتکل می‌تواند از آن برای تست پروتکل خود استفاده کند). گیرنده این قبیل فریمها را نادیده می‌گیرد، و هیچ عکس‌العملی به آنها نشان نمی‌دهد.

برخی از مهمترین گزینه‌هایی که می‌توان دربارهٔ آنها مذاکره کرد، عبارتند از: حداکثر اندازهٔ قسمت Payload فریم، فعال کردن احراز هویت و انتخاب پروتکل آن، فعال کردن مانیتورینگ کیفیت خط در طول عملیات، و انتخاب گزینه‌های فشرده‌سازی سرآیند.

دربارهٔ پروتکلهای NCP حرف کلی چندانی نمی‌توان زد. هر پروتکل NCP خاص یک نوع لایهٔ شبکه است، و بر سر گزینه‌های پیکربندی آن مذاکره می‌کند. برای مثال، در یک لایهٔ شبکهٔ IP احتمالاً مهمترین گزینه تخصیص آدرس IP است.

۳-۷ خلاصه

وظیفهٔ لایهٔ پیوند داده تبدیل استریم خام بیت‌های لایهٔ فیزیکی به استریمی از فریمها، و هدایت این استریم به لایهٔ شبکه است. روشهای مختلفی برای فریم‌بندی وجود دارد، که شمارش کاراکتر، لاگذاری بایت، و لاگذاری بیت از آن نمونه است. پروتکلهای لینک داده می‌توانند خطاها را کنترل کرده، و در صورت نیاز فریمهای معیوب را مجدداً ارسال کنند. برای جلوگیری از غرق شدن گیرنده‌های کُند در سیلاب داده‌های فرستنده‌های پُرسرعت، لایهٔ پیوند داده جریان داده‌ها را نیز کنترل می‌کند. یکی از پروتکلهایی که کنترل خطا و کنترل جریان در آن لحاظ شده، پروتکل پنجرهٔ لغزنده است.

پروتکلهای پنجرهٔ لغزنده را می‌توان بر حسب اندازهٔ پنجرهٔ ارسال و دریافت دسته‌بندی کرد. وقتی اندازهٔ این پنجره‌ها هر دو 1 باشد، پروتکل توقف-انتظار نام دارد. اگر پنجرهٔ ارسال و دریافت (برای اجتناب از قفل شدن فرستنده در محیطهای با تأخیر زیاد) بزرگتر از 1 باشند، گیرنده می‌تواند فریمهای خارج از نظم را بکلی دور انداخته و یا آنها را بافر کند.

در این فصل چندین پروتکل را مورد بررسی قرار دادیم. پروتکل 1 برای محیطهای عاری از خطا طراحی شده، و می‌تواند انتقال اطلاعات را با هر سرعتی انجام دهد. در پروتکل 2 محیط همچنان بدون خطا فرض شده، ولی قادر است جریان داده‌ها را کنترل کند. در پروتکل 3 برای کنترل خطا از شمارهٔ ترتیبی فریمها و الگوریتم توقف-انتظار استفاده شده است. ترافیک در پروتکل 4 می‌تواند دوطرفه باشد، و در آن تکنیکی بنام سواری مجانی معرفی شده است. پروتکل 5 از تکنیک پنجرهٔ لغزندهٔ «N تا به عقب برگرد» استفاده می‌کند. و بالاخره، در پروتکل 6 از تکنیکهای تکرار انتخابی و تصدیق دریافت منفی استفاده کردیم.

برای تست صحت و کارایی پروتکلها، روشهای مختلف مدلسازی (از جمله مدل ماشین حالت محدود و مدل شبکهٔ پتری) را معرفی کردیم.

شبکه‌های بسیاری در لایهٔ پیوند داده از پروتکلهای بیت-گرا استفاده می‌کنند، که از آن میان می‌توان به SDLC، HDLC، ADCCP، یا LAPB اشاره کرد. در تمام این پروتکلها ابتدا و انتهای فریم با یک بایت پرچم مشخص می‌شود، و اگر این بایت در داده‌ها موجود باشد، از تکنیک لاگذاری بیت برای مشخص کردن آن استفاده می‌شود. همهٔ این پروتکلها از یک پنجرهٔ لغزنده برای کنترل جریان سود می‌برند. در اینترنت نیز PPP بعنوان پروتکل اصلی لایهٔ پیوند داده در خطوط نقطه-به-نقطه کاربرد گسترده‌ای دارد.

مسائل

۱. یک بسته از لایهٔ بالاتر به ۱۰ فریم تقسیم شده، و احتمال اینکه هر یک از آنها سالم به مقصد برسد، ۸۰٪ است. اگر در لایهٔ پیوند داده کنترل خطا انجام نشود، این پیام چند بار باید فرستاده شود تا تمام آن صحیح و سالم به مقصد برسد؟

۲. در لایهٔ پیوند داده از کُدگذاری زیر استفاده شده است:

A: 01000111; B: 11100011; FLAG: 01111110; ESC: 11100000

در هر یک از حالتهای زیر، توالی بیت (باینری) فریم چهار کاراکتری A B ESC FLAG را نشان دهید:

(الف) شمارش کاراکتر.

(ب) بایت پرچم با لاگذاری بایت.

(ج) بایت پرچم در ابتدا و انتها، با لاگذاری بیت.

۳. با تکنیکی که در کتاب شرح داده شد، قطعه دادهٔ A B ESC C ESC FLAG FLAG D در وسط یک استریم ظاهر شده است. خروجی لاگذاری این قطعه چیست؟

۴. یکی از همکلاسی‌های شما عقیده دارد که قرار دادن یک بایت پرچم در ابتدای فریم و یکی در انتهای آن کار بیهوده‌ایست، و یک بایت کاملاً کفایت می‌کند (این بایت انتهای یک فریم، و ابتدای فریم بعدی محسوب خواهد شد). نظر شما چیست؟

۵. می‌خواهیم رشتهٔ 0111101111101111110 را از طریق لایهٔ پیوند داده ارسال کنیم. این رشته بعد از لاگذاری بیت به چه شکلی در می‌آید؟

۶. آیا در هنگام استفاده از لاگذاری بیت احتمال دارد که خطایی (از قبیل اضافه، کم شدن و یا تغییر یک بیت) رخ دهد، ولی جمع تطبیقی آنرا کشف نکند؟ اگر خیر، چرا؟ اگر بلی، چگونه؟ آیا طول جمع تطبیقی در اینجا نقشی دارد؟

۷. آیا فکر می‌کنید حالتی وجود دارد که یک پروتکل حلقهٔ-باز (مانند کُد همینگ) بر پروتکلهای مبتنی بر بازخور (که در این فصل دیدید) ارجحیت داشته باشد؟

۸. برای اطمینان بیشتر، بجای یک بیت توازن از کُدی استفاده می‌کنیم که یک بیت توازن برای بیت‌های فرد و یک بیت توازن دیگر برای بیت‌های زوج دارد. فاصلهٔ همینگ این کُد چقدر است؟

۹. برای ارسال یک پیام ۱۶ بیتی از کُد همینگ استفاده کرده‌ایم. برای آنکه گیرنده بتواند تمام خطاهای تک‌بیتی را کشف و تصحیح کند، به چند بیت چک‌کننده نیاز داریم. روش کار را برای پیام 1101001100110101 نشان دهید. فرض کنید در این کُد همینگ از توازن زوج استفاده شده است.

۱۰. می‌خواهیم یک بایت ۸ بیتی با مقدار باینری 10101111 را با استفاده از روش همینگ توازن زوج کُد کنیم. خروجی چیست؟

۱۱. یک کُد همینگ ۱۲ بیتی با مقدار هگزادسیمال 0xE4F به گیرنده می‌رسد. مقدار هگزادسیمال اولیه چه بوده است؟ فرض کنید بیش از یک بیت خطا رخ نداده است.

۱۲. یکی از تکنیکهای تشخیص خطا، ارسال داده‌ها بصورت ماتریسی از n سطر و k ستون است که هر سطر و ستون دارای بیت‌های توازن خاص خود است. آخرین بیت در منتهی‌الیه سمت راست بیتی است که سطر و ستون مربوطه را چک می‌کند. آیا این روش می‌تواند خطاهای تک‌بیتی را کشف کند؟ خطاهای دوبیتی را چطور؟ خطاهای سه‌بیتی را چطور؟

۱۳. ماتریسی با n سطر و k ستون دارای بیت‌های توازن افقی و عمودی است. فرض کنید در هنگام انتقال اطلاعات ۴ بیت تغییر کرده است. احتمال کشف نشدن این خطا را بصورت یک عبارت ریاضی استخراج کنید.

۱۴. حاصل تقسیم $x^7 + x^5 + 1$ بر چندجمله‌ای مولد $x^3 + 1$ چیست؟

۱۵. می‌خواهیم با استفاده از تکنیک CRC استریم 10011101 را ارسال کنیم. چندجمله‌ای مولد $x^3 + 1$ است. استریم فرستاده شده چیست؟ فرض کنید بیت سوم از سمت چپ در حین ارسال معکوس می‌شود. نشان دهید که گیرنده می‌تواند این خطا را کشف کند.

۱۶. پروتکلهای لینک داده همیشه CRC را بجای سرآیند در پی‌آیند پیام قرار می‌دهند. چرا؟

۱۷. یک کانال 4-kbps دارای تأخیر انتشار 20 msec است. تا چه اندازهٔ فریمی کارایی پروتکل توقف-انتظار بیش از ۵۰٪ است؟

۱۸. یک ترانک T1 بطول 3000 km از فریمهای 64 بایتی و پروتکل 5 استفاده می‌کند. اگر تأخیر انتشار 6μsec/km باشد، تعداد بیتهای شمارهٔ ترتیبی چقدر باید باشد؟

۱۹. آیا در پروتکل 3 فرستنده می‌تواند تایمری را که در حال کار است، از نو شروع کند؟ اگر بلی، در چه موقعیتی؟ اگر خیر، چرا؟

۲۰. فرض کنید در یک پروتکل پنجرهٔ لغزنده تعداد بیتهای شمارهٔ ترتیبی آنقدر زیاد است که «برگشت»

هرگز رخ نمی‌دهد. چه رابطه‌ای باید بین لبه‌های پنجره‌ها و اندازهٔ پنجره (که ثابت، و در فرستنده و گیرنده یکی است) برقرار باشد؟

۲۱. اگر روال *between* در پروتکل 5 بجای $a \le b < c$ شرط $a \le b \le c$ را چک کند، چه تأثیری روی درستی یا کارایی پروتکل خواهد گذاشت؟ توضیح دهید.

۲۲. در پروتکل 6 وقتی یک فریم به گیرنده می‌رسد، چک می‌کند که آیا شمارهٔ ترتیبی آن همانی است که باید باشد، و آیا *no_nak* مقدار true دارد یا خیر. اگر هر دو شرط true باشند، یک NAK فرستاده می‌شود؛ در غیر اینصورت، تایمر کمکی راه‌اندازی می‌شود. اگر قسمت else را حذف کنیم، چه تأثیری روی درستی پروتکل می‌گذارد؟

۲۳. فرض کنید حلقهٔ while سه دستوری نزدیک به انتهای پروتکل 6 را حذف کرده‌ایم. آیا این کار بر درستی پروتکل اثر می‌گذارد، یا فقط کارایی آنرا تحت تأثیر قرار می‌دهد؟ توضیح دهید.

۲۴. فرض کنید بخش case مربوط به خطاهای جمع تطبیقی را از دستور switch پروتکل 6 حذف کرده‌ایم. این کار چه تأثیری بر عملکرد پروتکل خواهد گذاشت؟

۲۵. کُد *frame_arrival* در پروتکل 6 بخشی برای NAK ها دارد. این بخش زمانی اجرا می‌شود که فریم ورودی یک NAK باشد و شرط دیگری وجود داشته باشد. سناریویی را شرح دهید که در آن وجود این شرط دوم الزامی باشد.

۲۶. فرض کنید می‌خواهید برای لایهٔ پیوند داده خطی برنامه بنویسید که در آن داده‌ها فقط به سمت شما می‌آیند، و شما هیچ چیزی نمی‌فرستید. سمت مقابل از پروتکل HDLC با شمارهٔ ترتیبی ۳ بیتی، و پنجرهٔ ۷ فریمی استفاده می‌کند. برای بالا بردن کارایی سیستم، تصمیم گرفته‌اید حداکثر فریمهای خارج از نظم ممکنه را بافر کنید، ولی مجاز به دستکاری در نرم‌افزار سمت فرستنده نیستید. آیا می‌توان پنجرهٔ دریافتی بزرگتر از 1 داشت، و تضمین کرد که این پروتکل هرگز با شکست مواجه نشود؟ اگر بلی، بزرگترین پنجره‌ای که می‌توان با اطمینان بکار برد، چقدر است؟

۲۷. پروتکل 6 را روی یک خط 1-Mbps عاری از خطا در نظر بگیرید. حداکثر اندازهٔ فریم 1000 بیت است. بسته‌ها با فواصل یک ثانیه‌ای تولید می‌شوند. فاصلهٔ زمانی انقضای تایمر 10 msec است. اگر تایمر مخصوص تصدیق دریافت را حذف کنیم، انقضاهای غیرلازم رخ خواهد داد. یک پیام با طول متوسط چند بار باید مجدداً ارسال شود؟

۲۸. در پروتکل 6 داریم: $MAX\_SEQ = 2^n - 1$. با اینکه این شرط برای استفادهٔ بهینه از بیت‌های سرآیند آشکارا مناسب است، نشان ندادیم که الزامی هم هست. اگر، برای مثال $MAX\_SEQ = 4$ ، آیا باز هم این پروتکل بدرستی کار می‌کند؟

۲۹. با استفاده از یک کانال ماهواره‌ای 1-Mbps که زمان رسیدن سیگنال از زمین به ماهواره 270 msec است، فریمهای 1000 بیتی ارسال می‌کنیم. فریمهای تصدیق دریافت همیشه با سواری مجانی برمی‌گردند، و سرآیند فریمها بسیار کوتاه است. حداکثر نرخ مصرف قابل‌دستیابی در پروتکلهای زیر چقدر است؟

(الف) پروتکل توقف-انتظار.

(ب) پروتکل 5 .

(ج) پروتکل 6 .

۳۰. مقدار اتلاف پهنای باند پروتکل 6 را (مربوط به سرآیند و ارسال مجدد) در یک کانال ماهواره‌ای 50-kbps پُرترافیک، با فریمهایی متشکل از 40 بیت سرآیند و 3960 بیت داده، محاسبه کنید. فرض کنید زمان رسیدن سیگنال از زمین به ماهواره 270 msec است؛ فریمهای ACK هرگز ارسال نمی‌شوند؛

فریمهای NAK دارای طولی معادل 40 بیت هستند؛ نرخ خطا در فریمهای داده ۱٪ ، و در فریمهای NAK قابل چشم‌پوشی است؛ و شماره‌های ترتیبی 8 بیتی هستند.

۳۱. می‌خواهیم روی یک کانال ماهواره‌ای عاری از خطا با ظرفیت 64 kbps فریمهای 512 بایتی (در یک جهت)، با فریمهای تصدیق دریافت بسیار کوتاه (در جهت دیگر)، بفرستیم. اگر اندازهٔ پنجره 1 ، 7 ، 15 و 127 باشد، حداکثر ظرفیت خط چقدر خواهد بود؟ زمان ارسال سیگنال از زمین به ماهواره را 270 msec بگیرید.

۳۲. یک کابل T1 بطول 100 km را در نظر بگیرید. سرعت انتشار امواج در این کابل 2/3 سرعت نور در خلاء است. چند بیت در این کابل جا می‌شود؟

۳۳. فرض کنید پروتکل ۴ را با ماشین حالت محدود مدل کرده‌ایم. هر ماشین چند حالت دارد؟ کانال مخابراتی چند حالت دارد؟ سیستم کامل (دو ماشین و یک کانال مخابراتی) چند حالت دارد؟ از خطاهای جمع تطبیقی صرفنظر کنید.

۳۴. توالی آتش شبکهٔ پتری شکل ۳-۲۳ که با توالی حالت (01A), (010), (01-), (01A), (000) در شکل ۳-۲۱ متناظر است، را در نظر بگیرید. توضیح دهید که این توالی چه چیزی را نشان می‌دهد.

۳۵. شبکهٔ پتری دستورات گذار $AC \rightarrow B$ ، $AC \rightarrow B$ ، $CD \rightarrow E$ ، و $E \rightarrow CD$ را رسم کنید. از این شبکهٔ پتری گراف دسترسی حالت محدود برای رسیدن به حالت ACD را رسم کنید. این گراف چه مفهوم شناخته‌شده‌ای را مدل می‌کند؟

۳۶. پروتکل PPP از HDLC مشتق شده، که در آن برای جلوگیری از تفسیر اشتباه بایت پرچم در داده‌ها از تکنیک لاگذاری بیت استفاده می‌شود. یک دلیل بیاورید که چرا PPP از لاگذاری بایت استفاده می‌کند، نه لاگذاری بیت.

۳۷. حداقل سربارهٔ یک بستهٔ IP که با PPP فرستاده شده، چقدر است؟ فقط سربارهٔ PPP را در نظر بگیرید، نه سربارهٔ سرآیند IP را.

۳۸. هدف از این تمرین آزمایشگاهی پیاده‌سازی یک مکانیزم کشف خطا با استفاده از الگوریتم CRC (که در متن کتاب توضیح دادیم) است. دو برنامه بنویسید: مولد (generator) ، و تست‌کننده (verifier) . برنامهٔ مولد یک رشتهٔ n بیتی از 0 ها و 1 ها را از ورودی استاندارد (با فرمت ASCII) می‌خواند. خط دوم یک چندجمله‌ای k بیتی است، که آن هم بصورت ASCII از ورودی خوانده می‌شود. برنامه خروجی خود را، که تشکیل شده از $n + k$ بیت 0 و 1 (و در واقع همان ورودی کُد شده است)، روی خروجی استاندارد می‌نویسد. (برنامهٔ مولد این خروجی را هم با فرمت ASCII بیرون می‌دهد.) برنامهٔ تست‌کننده خروجی مولد را گرفته، و با پیامی نشان می‌دهد که آیا این رشته درست است یا خیر. برای تست این دو برنامه، یک برنامهٔ کمکی دیگر (بنام alter) بنویسید که یکی از بیتهای خط اول را معکوس کند و سایر بیت‌ها را بهمان صورت باقی بگذارد (شمارهٔ بیتی که باید معکوس شود را - با آغاز شمارش از سمت چپ - بصورت آرگومان ورودی به این برنامه بدهید). با نوشتن

```
generator < file | verifier
```

برنامه باید پاسخ «درست است» بدهد، اما با دستور

```
generator < file | alter arg | verifier
```

باید پیام «درست نیست» بگیرید.

۳۹. برنامه‌ای بنویسید که رفتار یک شبکهٔ پتری را شبیه‌سازی کند. این برنامه باید دستورات گذار، و لیستی از حالتهای لایهٔ شبکه (هنگام ارسال و دریافت یک بستهٔ جدید) را بعنوان ورودی بخواند. برنامه باید از حالت اولیه (که آنرا هم از ورودی می‌خواند) یکی از گذارهای فعال را بصورت تصادفی آتش کند، و سپس چک کند که آیا هر یک از ماشینها ۲ بسته قبول می‌کند، بدون این که بین آنها یک بستهٔ جدید بفرستد.

زیر لایه نظارت بر دسترسی به رسانه انتقال

به گونه‌ای که در فصل اوّل اشاره کردیم، شبکه‌ها را می‌توان به دو رده تقسیم بندی کرد: آنهائی که از اتصالات نقطه‌به‌نقطه استفاده می‌کنند و آنهائی که از کانالهای پخش فراگیر (Broadcast) بهره می‌گیرند. در این فصل به شبکه‌های پخش فراگیر و پروتکلهای آنها خواهیم پرداخت.

در هر شبکهٔ فراگیر مسئله اصلی آنست که وقتی برای دسترسی به کانال انتقال ، رقابت وجود دارد چگونه می‌توان تعیین کرد که چه کسی باید از کانال استفاده کند. برای روشن‌تر شدن قضیه یک کنفرانس تلفنی را با حضور شش نفر، در نظر بگیرید که از طریق شش خط تلفن بهم متصل شده و نشستی را ترتیب داده‌اند و هر کدام قادرند با دیگران گفت و شنود داشته باشند. در چنین وضعیتی کاملاً طبیعی است که وقتی یک نفر صحبت خود را قطع می‌کند دو یا چند نفر به طور همزمان شروع به حرف زدن نمایند و برای لحظاتی بی‌نظمی و اغتشاش بر جلسه حاکم شود. در ملاقاتهای رو در رو بکمک اشاره و علامت، از بروز بی‌نظمی احتراز می‌شود؛ مثلاً شخص مایل به گفتگو، دست خود را به علامت اجازه خواستن برای شروع سخن، بالا می‌برد. وقتی فقط یک کانال منفرد در اختیار است، تعیین نفر بعدی برای ارسال، دشوارتر خواهد بود. برای حل این مسئله پروتکلهای متعددی عرضه شده است که معرفی آنها شاکلهٔ این فصل را تشکیل می‌دهد. در ادبیات شبکه، کانالهای فراگیر گاهی با عناوین «**کانالهای با دسترسی چندگانه**» (Multiaccess Channel) یا «**کانالهای با دسترسی تصادفی**» (Random Access Channel) معرفی می‌شوند.

پروتکلهائی که برای تعیین نفر بعدی در استفاده از کانال مشترک کاربرد دارند متعلق به زیرلایه‌ای از لایهٔ پیوند داده‌ها هستند که اصطلاحاً زیرلایهٔ MAC (Medium Access Control) نامیده می‌شود. زیرلایهٔ MAC در شبکه‌های محلی که اغلب آنها از کانالهای مشترک به عنوان زیربنای ارتباط استفاده می‌کنند، از اهمیت ویژه‌ای برخوردار است. در مقابل، به غیر از شبکه‌های ماهواره‌ای ، تمام شبکه‌های WAN از خطوط نقطه‌به‌نقطه بهره گرفته‌اند [و در آنها زیرلایهٔ MAC جایگاهی ندارد]. از آنجایی که کانالهای با دسترسی چندگانه و شبکه‌های LAN کاملاً به یکدیگر مرتبط هستند لذا در این فصل بطور عامّ شبکه‌های LAN را بررسی می‌کنیم و موارد معدودی را نیز که مستقیماً مربوط به زیرلایهٔ MAC نمی‌شوند، بررسی خواهیم نمود.

از دیدگاه فنّی، زیرلایهٔ MAC بخش زیرین لایهٔ پیوندِ داده‌ها محسوب می‌شود و منطقاً می‌بایست قبل از آنکه در فصل سوم به پروتکل‌های نقطه‌به‌نقطه بپردازیم ابتدا این زیرلایه را بررسی می‌کردیم. ولیکن برای اغلب افراد ، فهم پروتکلهای دوطرفه ساده‌تر از پروتکل‌های چندطرفه است.[منظور از پروتکل دوطرفه پروتکلیست که در آن فقط و فقط دو ماشین درگیر مبادله داده هستند.] به همین دلیل در روند تشریح مطالب (که از لایه‌های پائین به

سمت بالا خواهد بود) ، اندکی از این ترتیب تخطی کردیم.

۱ـ٤ مسئله تخصیص کانال

مضمون اصلی این فصل آنست که چگونه یک کانال مشترک و فراگیر را بین کاربران رقیب تقسیم نمائیم. در ابتدا به الگوهای تخصیص «ایستا» و «پویا» نگاهی می‌اندازیم؛ سپس به تشریح تعدادی از الگوریتمهای خاصّ در این زمینه خواهیم پرداخت.

۱ـ۱ـ٤ تخصیص ایستای کانال در شبکه‌های LAN و MAN

روش سنّتی در تخصیص یک کانال منفرد ، (مثل خطوط اصلی تلفن – Telephone Trunk) ، بین چندین کاربر که برای استفاده از آن رقابت می‌کنند ، روش FDM (تسهیم در حوزهٔ فرکانس) است. اگر N کاربر حضور داشته باشند، پهنای باند کانال به N بخش متساوی تقسیم می‌شود و به هر کاربر یکی از این بخشها اختصاص داده می‌شود (شکل ۲ـ۳۱ را ببینید). از آنجایی که هر کاربر دارای یک باند فرکانس اختصاصی است لذا کاربران هیچگونه تداخل و مزاحمتی برای یکدیگر ندارند. وقتی تعداد کاربران ثابت و کم باشد و هر کدام نیز دارای بار سنگین (و بافر شدهٔ) ترافیک باشند (همانند مراکز سوئیچ تلفن) ، روش FDM مکانیزمی ساده و کارآمد برای تخصیص کانال خواهد بود. ولیکن وقتی تعداد ارسال کنندگان زیاد و دائماً در حال تغییر باشد ، یا ترافیک ارسالی آنها به صورت لحظه‌ای و انفجاری تولید شود، FDM مشکلات متعددی را بروز خواهد داد: اگر طیف فرکانسی کانال به N بخش مجزا تقسیم شود ولی تعداد کاربرانی که تمایل به ارسال و مخابرهٔ داده دارند کمتر از N باشد بخش ارزشمندی از این طیف فرکانسی تلف خواهد شد. اگر تعداد کاربران بیش از N باشد، برخی از آنها بدلیل کمبود پهنای باند ، مجوز ارسال نخواهند داشت؛ حتی اگر برخی از کاربرانی که بدانها باند فرکانسی تخصیص داده شده به ندرت بخواهند چیزی ارسال یا دریافت کنند.

با این وجود ، حتی اگر بتوان تعداد کاربران را عدد ثابت N فرض کرد ، باز هم تقسیم ایستای طیف فرکانسی کانال منفرد به تعدادی زیرکانال ، ذاتاً روشی ناکارآمد و بی‌کفایت تلقی می‌شود. مشکل اساسی آنست که وقتی برخی از کاربران ، تقاضای ارسال نداشته باشند پهنای باند آنها هدر می‌رود. گذشته از آن در بسیاری از سیستم‌های کامپیوتری، ترافیک داده‌ها بشدّت انفجاری است (نسبت حداکثر ترافیک به متوسط ترافیک در حدود 1000:1 است). در نتیجه اکثر اوقات کانالها خالی و بلااستفاده باقی می‌مانند.

کارآئی بسیار ضعیفِ روش FDM را می‌توان با یک محاسبهٔ ساده در «**نظریهٔ صفّ**» (Queuing Theory) اثبات کرد. برای شروع ، فرض کنید بخواهیم زمان متوسط تاخیر T را برای کانالی محاسبه کنیم که در آن نرخ ارسال C بیت بر ثانیه ، نرخ دریافت فریمها λ فریم بر ثانیه و طول هر فریم تصادفی است و از تابع چگالی احتمال نمائی با میانگین $1/\mu$ بیت بر فریم تبعیت می‌کند.

با این پارامترها، نرخ دریافت فریمها λ فریم بر ثانیه و نرخ سرویس‌دهی $\mu.c$ فریم بر ثانیه خواهد بود. با استفاده از نظریهٔ صفّ ، می‌توان نشان داد که اگر نرخ دریافت و زمان سرویس‌دهی به فریمها از تابع توزیع پوآسون تبعیت کند خواهیم داشت:

$$T = \frac{1}{\mu.C-\lambda}$$

بعنوان مثال اگر C معادل 100 مگابیت بر ثانیه ، متوسط طول فریمها (یعنی $1/\mu$) معادل 10000 بیت و نرخ دریافت فریمها (یعنی λ)معادل 5000 فریم بر ثانیه باشد، متوسط تاخیر هر فریم $T = 200\,\mu s$ خواهد بود. دقت کنید که اگر از تاخیر صفّ (یعنی تاخیر انتظار فریم) صرف‌نظر می‌کردیم و زمان ارسال 10000 بیت را بر روی یک شبکهٔ

100Mbps محاسبه می‌نمودیم به پاسخ اشتباهِ 100 میکروثانیه می‌رسیدیم. این نتیجه فقط زمانی صدق می‌کند که هیچ رقابتی در بدست گرفتن کانال وجود نداشته باشد. حال بیائید این کانال منفرد را به N زیرکانال مستقل با ظرفیت C/N بیت بر ثانیه تقسیم کنیم. در این حالت نرخ متوسط ورودی به هر یک از این زیرکانالها λ/N خواهد بود. با محاسبه مجددِ T به دست می‌آوریم:

$$T_{FDM} = \frac{1}{\mu(C/N) - (\lambda/N)} = \frac{N}{\mu.C-\lambda} = N.T$$ رابطهٔ (۴-۱)

متوسط تاخیر در FDM ، N برابر بیشتر از زمانی است که تمام فریمها به ترتیب در یک صفّ مرکزی طولانی و مرتب شده، پشت سرهم ارسال شوند.

تمام استدلالات و مباحثاتی که در مورد روش FDM اعمال کردیم دقیقاً در مورد TDM نیز صدق می‌کند. به هر کاربر بصورت ثابت یکی از N برش زمانی (slot time) اختصاص داده می‌شود. اگر یک کاربر از برش زمانیِ تخصیص یافته به خود استفاده نکند ، آن زمان بلااستفاده مانده و هدر خواهد رفت. با استفادهٔ مجدد از مثال قبلی، اگر یک شبکهٔ 100Mbps را به ده شبکهٔ 10Mbps تقسیم کرده و هر کاربر از یکی از آنها استفاده کند، میانگین تاخیر از 200 میکروثانیه به 2 میلی‌ثانیه افزایش خواهد یافت.

از آنجاکه هیچیک از روشهای معمول و ایستای تخصیص کانال در محیطهای با ترافیک انفجاری کار نخواهد کرد لذا در ادامه به بررسی روشهای پویا می‌پردازیم.

۴-۱-۲ تخصیص پویای کانال در LAN و MAN

قبل از آنکه به اوّلین روش از روشهای متعدّدِ تخصیص کانال بپردازیم دسته‌بندی و فرموله کردن مسائل و مشکلات تخصیص کانال مفید خواهد بود. کل کاری که باید برای تخصیص کانال انجام شود مبتنی بر پنج فرض اساسی است که در زیر تشریح شده‌اند:

۱. **مدل ایستگاه (Station Model):** این مدل شامل N ایستگاهِ مستقل (مثل کامپیوتر، تلفن یا دستگاه‌های مخابرهٔ شخصی) است که در هر کدام از آنها یک برنامه یا کاربر ، فریمهایی را برای ارسال تولید می‌کند. برخی از اوقات به ایستگاه‌ها، «پایانه» (یا ترمینال) نیز گفته می‌شود. احتمال آنکه در بازهٔ زمانی ΔT فریمی تولید شود، $\lambda.\Delta T$ است که در آن λ یک مقدار ثابت است (در حقیقت λ میانگین نرخ تولید فریمهای جدید است).به محض آنکه فریمی تولید گردد ، ایستگاه متوقف شده و تا زمانی که آن فریم به صورت موفقیت‌آمیز ارسال نشود کاری انجام نمی‌دهد.

۲. **فرض کانال منفرد (Single Channel Assumption):** در این حالت تنها یک کانال منفرد و مشترک برای مخابرهٔ داده در اختیار ایستگاه‌ها است. تمام ایستگاه‌ها می‌توانند اطلاعات خود را برروی این کانال بفرستند یا از آن دریافت کنند. از دیدگاه سخت‌افزاری تمام ایستگاه‌ها هم‌ارز و معادل یکدیگرند؛ اگرچه ممکن است در نرم‌افزارِ پروتکل به هر ایستگاه ، اولویتی خاصّ داده شود.

۳. **فرض تصادم (Collision Assumption):** هرگاه دو فریم بطور همزمان [برروی کانال مشترک] ارسال شوند با یکدیگر تداخل کرده و سیگنال حاصل بی‌ارزش و نامعتبر خواهد بود. این رخداد اصطلاحاً «**تصادم**» (Collision) نامیده می‌شود. هر ایستگاه می‌تواند از وقوع تصادم آگاه گردد. فریمی که در حین ارسال آن تصادم رخ داده است باید از نو فرستاده شود. در این مدل هیچ خطائی به غیر از خرابی فریم در اثر تصادم لحاظ نمی‌شود.

۴.*الف*: **مدل زمان پیوسته (Continuous Time):** در این مدل، ارسال فریمها می‌تواند در هر لحظه از زمان

شروع شود و هیچگونه سیگنال ساعت مرکزی (سراسری) که زمان را به برشهای گسسته و مجزا تقسیم کند، وجود ندارد.

۴.ب: **«مدل زمان گسسته» (Slotted time):** در این مدل، زمان به برشهای گسسته و مستقلی تقسیم می‌شود و ارسال فریم همیشه باید در ابتدای یکی از این برشهای زمانی انجام گیرد. در هر برش زمان ممکن است صفر، یک، یا چند فریم ارسال شود که به ترتیب: صفر فریم به معنای بیکار و بلااستفاده ماندن آن برش زمانی، یک فریم به معنای ارسال موفق و چند فریم معادل تصادم خواهد بود.

۵.الف: **شنود سیگنال حامل (Carrier Sense):** در این مدل، ایستگاه‌ها قبل از شروع به ارسال فریم خود، قادرند تشخیص بدهند که آیا کانال مشغول است یا آزاد؟ اگر ایستگاهی احساس کند که کانال مشغول است هیچگاه سعی در استفاده از آن نخواهد کرد مگر آنکه مجدداً کانال بیکار شود.

۵.ب: **عدم شنود سیگنال حامل (No Carrier Sense):** در این مدل، ایستگاه‌ها قادر نیستند قبل از استفاده از کانال، آنرا بشنوند و سیگنال روی آنرا احساس کنند؛ لذا فقط پس از فرستادن فریم می‌توان تعیین کرد که آیا ارسال موفق بوده یا تصادم پدید آمده است.

اندکی توضیح در خصوص برخی از فرضیات فوق مفید خواهد بود: اوّلین بند اذعان می‌دارد که ایستگاه‌ها مستقل هستند و فریمها را با نرخ ثابتی تولید می‌کنند. [یعبارتی نرخ میانگین تولید فریمها ثابت است. -م] همچنین در این بند تلویحاً فرض شده که هر ایستگاه تنها یک کاربر یا برنامهٔ فعال دارد و بدین ترتیب هرگاه یک ایستگاه، متوقف (بلوکه) شود هیچ فریم جدیدی تولید نخواهد شد. مدلهای پیچیده دیگری نیز وجود دارد که در آنها ایستگاه‌ها اجازه دارند به صورت چندبرنامه‌ای (Multiprogrammed) تولید فریمها را ادامه بدهند (حتی وقتی که ایستگاه در انتظار ارسال فریم قبلی بلوکه شده است)؛ ولیکن تحلیل چنین ایستگاه‌هائی بسیار دشوار است.

فرض **کانال منفرد**، هستهٔ اصلی این مدل است. ایستگاه‌ها بجز یک کانال واحد و مشترک راهی برای مبادلهٔ اطلاعات ندارند. ایستگاه‌ها نمی‌توانند همانند یک کلاس درس با بالا بردن دست خود، از معلم کلاس برای صحبت کردن اجازه بگیرند!

فرض **تصادم** نیز محوری است، اگر چه برخی از سیستمها (بویژه سیستمهای مبنی بر «طیف گسترده» - Spread Spectrum) از این فرض مستثنی بوده و نتایج شگفت‌آوری نیز به همراه دارند؛ همچنین در شبکه‌های توکن‌رینگ (Token Ring) یک نشانهٔ خاص (توکن) ایستگاه به ایستگاه می‌چرخد و هر ایستگاه که آنرا در اختیار بگیرد اجازهٔ ارسال فریم خود را خواهد داشت ولیکن در بخشهای بعدی به کانالهای منفردی خواهیم پرداخت که برای استفاده از آنها رقابت وجود دارد و تصادم پدید می‌آید.

در خصوص زمان ارسال فریمها، دو فرض (۱) زمان پیوسته (۲) زمان گسسته قابل اعمال است. برخی از سیستمها از مدل اوّل پیروی می‌کنند و برخی دیگر از مدل دوّم؛ بنابراین ما هر دو مدل را تحلیل خواهیم کرد. در هر سیستم فقط یکی از این مدلها قابل اعمال است.

همچنین در یک شبکه ممکن است سیگنال روی کانال احساس شود (فرض ۵-الف) یا شنود سیگنال میسر نباشد (فرض ۵-ب). شبکه‌های محلی (LAN) عموماً قادر به احساس سیگنال روی کانال هستند ولیکن در شبکه‌های بی‌سیم این مدل قابل استفاده نخواهد بود چراکه برخی از ایستگاه‌ها در محدودهٔ شنود سیگنال ایستگاه‌های دیگر نیستند. ایستگاه‌هائی که به کانالهای سیمی متصل هستند در مدل «شنود سیگنال حامل» (Carrier Sense) قرار می‌گیرند و قادرند به محض کشف پدیدهٔ تصادم، ارسال فریم را خاتمه بدهند. کشف تصادم در شبکه‌های بی‌سیم به دلایل فنی‌مهندسی به ندرت قابل انجام است. دقت کنید که کلمهٔ «حامل»

(Carrier) در اینجا اشاره به یک سیگنال الکتریکی برروی کابل دارد.

۲-٤ پروتکلهای دسترسی چندگانه

الگوریتم‌های بیشماری در مورد تخصیص کانالهای یا دسترسی چندگانه [کانالهای مشترک] معرفی شده‌اند. در بخشهای آتی نمونه‌هائی از جالبترین این پروتکلها را بررسی کرده و مثالهایی از کاربرد آنها ارائه خواهیم نمود.

۱-۲-٤ ALOHA

در آوان ۱۹۷۰، نورمن آبرامسون و همکاران او در دانشگاه هاوائی روشی جدید و جالب برای حل مسئله تخصیص و دسترسی به کانال‌های اشتراکی ابداع کردند. بعداً، کار آنها توسط بسیاری از پژوهشگران ادامه یافت و تکمیل شد. (مرجع Abramson, 1985) اگر چه کار آقای آبرامسون که سیستم ALOHA نامیده شد براساس پخش امواج رادیوی زمینی بود ، ولیکن نظریه آنها در هر سیستمی که در آن کاربران برای استفاده از یک کانال مشترک ، به صورت ناهماهنگ رقابت می‌کنند، قابل اعمال و پیاده‌سازی است.

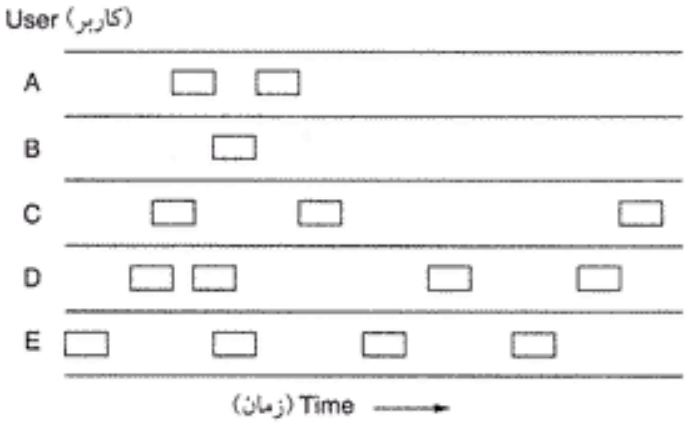
در اینجا دو نسخه متفاوت از ALOHA یعنی Pure ALOHA و Slotted ALOHA را تشریح خواهیم کرد. تفاوت این دو نسخه در تقسیم‌بندی زمان به برشهائی است که فریمها بتوانند در خلال یکی از آنها ارسال شوند. Pure ALOHA نیازی به هماهنگی زمانی (Time Synchronization) ندارد در حالیکه Slotted ALOHA نیازمند این هماهنگی است.

Pure ALOHA

ایده اصلی در سیستم ALOHA بسیار ساده است: کاربران اجازه دارند هر زمان که داده‌ای برای ارسال داشتند آنرا بفرستند. البته تصادمهایی رخ خواهد داد و فریمهایی که تصادم کنند از بین خواهند رفت. با این وجود در ALOHA ، یک «کانال بازگشت سیگنال» (Feedback) وجود دارد که فرستنده با گوش دادن به این کانال ، می‌تواند متوجه بروز تصادم و خرابی فریم شود. در شبکه‌های محلی LAN بروز تصادم به صورت سریع و آنی کشف می‌شود در حالیکه در شبکه‌های ماهواره‌ای یک ایستگاه پس از گذشت ۲۷۰ میلی ثانیه می‌تواند متوجه شود که آیا ارسال او موفق بوده یا نه! هرگاه به هر دلیلی امکان شنود سیگنال بازگشتی در حین ارسال وجود نداشته باشد بایستی دریافت فریمها توسط گیرنده تایید گردد. [با ارسال فریمهای مستقلی به نام Ack] هرگاه فریمی خراب شود، فرستندهٔ آن ، به اندازه یک زمان تصادفی صبر خواهد کرد و آن را مجدداً ارسال می‌کند. زمان انتظار قطعاً باید تصادفی باشد وگرنه تصادمها در یک دور نامتناهی تکرار خواهند شد. سیستمهایی که در آنها چندین کاربر از یک کانال مشترک به نحوی استفاده کنند که احتمال تصادم و تلاقی وجود دارد اصطلاحاً «**سیستم‌های رقابتی**» نامیده می‌شوند. در شکل ۴-۱ نمایشی از تولید فریم در سیستم ALOHA نشان داده شده است. در این شکل طول تمام فریمها را یکسان در نظر گرفته‌ایم زیرا اگر طول فریمها ، اندازهٔ ثابتی داشته باشند کارآئی و توان خروجی ALOHA حداکثر خواهد بود.

هرگاه دو فریم بطور همزمان برروی کانال ارسال شوند، تصادم رخ داده و هر دو خراب خواهند شد. حتی اگر اولین بیت از یک فریم جدید با آخرین بیت از فریم قبلی تداخل کند هر دو فریم بطور کامل خراب شده و بعداً باید از نو ارسال شوند زیراکدهای کشف خطا نمی‌توانند (و نباید هم بتوانند) تشخیص بدهند خطا در کجا اتفاق افتاده و بدین ترتیب کلّ فریم بلااستفاده خواهد بود.

سوال جالبی که پیش می‌آید آنست که کارآئی کانال در ALOHA چقدر است؟ بعبارت دیگر می‌خواهیم بدانیم در این محیطِ نامنظم و هرج‌ومرج ، چند درصد از کلّ فریمهای ارسالی از تصادم جان سالم به در می‌برند؟



شکل ۴-۱. در Pure ALOHA فریمها در زمانهای کاملاً دلخواه ارسال می‌شوند.

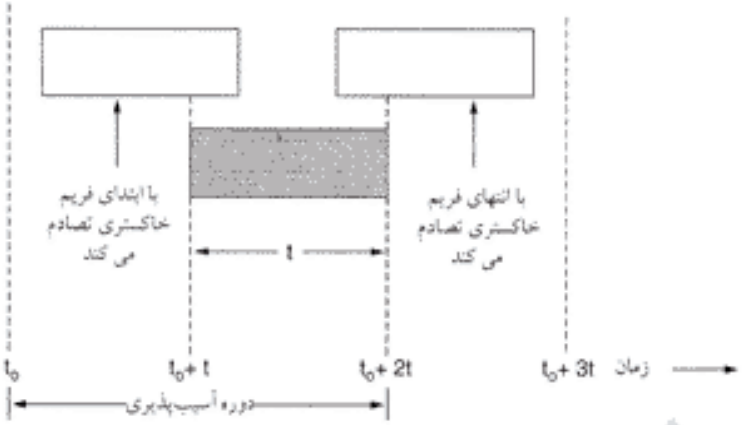
در ابتدا فرض می‌کنیم که تعدادی نامتناهی از کاربران در شبکه وجود دارند که پشت کامپیوترهای خود نشسته‌اند؛ هر کاربر در یکی از دو وضعیت «تایپ» یا «انتظار» قرار دارد. هرگاه تایپ یک خط به اتمام رسید و کلید Enter فشار داده شد، کاربر از تایپ دست می‌کشد و در انتظار پاسخ باقی می‌ماند. ایستگاه، فریم حاوی این خطّ را بلافاصله بر روی کانال می‌فرستد و بررسی می‌کند که آیا ارسال موفقیت‌آمیز بوده است؟ اگر فرآیند ارسال موفق باشد کاربر پاسخ خود را دریافت کرده و عمل تایپ را از سرمی‌گیرد و در غیر اینصورت، کاربر باز هم منتظر می‌ماند تا ارسال فریم آنقدر تکرار شود تا بالاخره یکی از آنها به سلامت به مقصد برسد.

«**زمان فریم**» (Frame Time)، مقدار زمانی است که طول می‌کشد تا یک فریم با طول ثابت و استاندارد ارسال شود. (به عبارت دیگر این زمان معادل با طول فریم تقسیم بر نرخ ارسال خواهد بود.) در اینجا فرض را بر آن می‌گذاریم که تعداد نامحدودی کاربر، مبتنی بر تابع توزیع پوآسون و با میانگین N فریم در واحد زمان، فریم‌های جدید تولید می‌کنند. (واحد زمان در اینجا زمان لازم برای ارسال یک فریم است). فرض بی‌نهایت بودن کاربران از آن جهت لازم است که مطمئن باشیم وقتی یک کاربر متوقف و منتظر می‌شود از تعداد کاربران کاسته نخواهد شد. اگر $N > 1$ باشد مجموع کاربران با نرخی بیشتر از ظرفیت کانال، فریم تولید کرده‌اند و تقریباً تمام فریمها در اثر تصادم نابود خواهند شد. برای آنکه کارآئی قابل ملاحظه‌ای داشته باشیم انتظار می‌رود که $0 < N < 1$ باشد. هر ایستگاه علاوه بر فریمهای جدید خود، فریمهایی را که قبلاً در اثر تصادم خراب شده‌اند، نیز ارسال می‌کند.

اجازه بدهید فرض کنیم که احتمال «تلاش برای ارسال k فریم در واحد زمان» نیز از تابع توزیع پوآسون با متوسط G تبعیت کند. (به خاطر داشته باشید که واحد زمان در اینجا زمان لازم جهت ارسال یک فریم -Frame Time- است.) بدیهی است که $G \geq N$. [زیرا N متوسط تولید فریم‌های جدید است در حالیکه G متوسط تولید فریم‌های جدید و فریم‌های قبلی خراب شده می‌باشد.-م]

در بار پائین (یعنی $N \approx 0$) تعداد تصادمها نیز اندک بوده و طبعاً ارسال مجدد فریمها ناچیز خواهد بود لذا ($G \approx N$) است. در بار بالا تصادمهای زیادی رخ می‌دهد یعنی $G > N$ است؛ در هر شرایط بار، بازده مفید کانال (یعنی S) مساوی است با حاصل ضرب میزان بار (یعنی G) در احتمال موفقیت در ارسال (یعنی P_0) بنابراین داریم: $S = G \times P_0$ که در آن P_0 احتمال **عدم خرابی** یک فریم در اثر تصادم است.

فقط وقتی یک فریم در اثر تصادم خراب نخواهد شد که در زمان ارسال آن هیچ فریم دیگری ارسال نشود؛ به شکل ۴-۲ دقت کنید. تحت چه شرایطی فریمی که در شکل بصورت سایه‌دار نشان داده شده است سالم به مقصد خواهد رسید؟ t را زمان لازم برای ارسال یک فریم در نظر بگیرید. هرگاه کاربر دیگری در فاصلهٔ زمانی t_0 تا t_0+t فریمی را تولید و ارسال کرده باشد انتهای فریم او با ابتدای فریم سایه‌دار تصادم خواهد کرد. در حقیقت سرنوشت



شکل ۴-۲. دورهٔ آسیب‌پذیری برای فریم خاکستری.

فریم سایه‌دار به گذشته نیز بستگی دارد، حتی قبل از آنکه اولین بیت آن ارسال شود؛ چراکه در سیستم Pure ALOHA، ایستگاه قبل از شروع به ارسال یک فریم قادر نیست به کانال گوش داده و متوجه شود که فریم دیگری در حال ارسال است. بدلیل مشابه اگر فریم جدیدی در فاصلهٔ زمان $t_0 + t$ تا $t_0 + 2t$ ارسال شود با انتهای فریم سایه‌دار در شکل تصادم خواهد کرد.

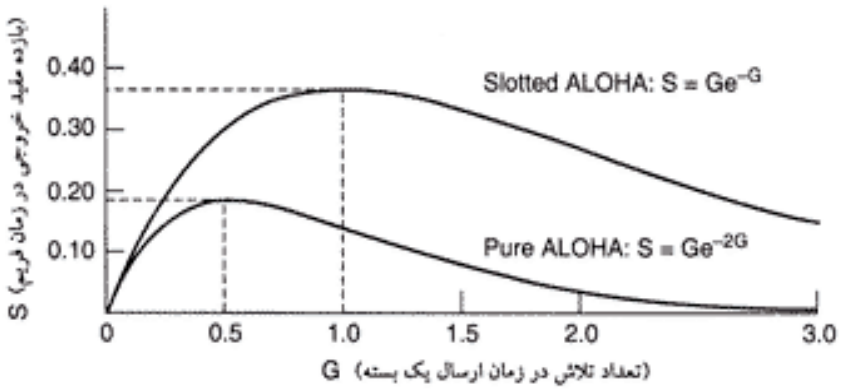
احتمال آنکه در زمان ارسال یک فریم [یعنی در زمان t] تعداد K فریم تولید شود از تابع توزیع پوآسون تبعیت می‌کند یعنی:

$$Pr[k] = \frac{G^k \cdot e^{-G}}{k!}$$ رابطهٔ (۴-۲)

بدین ترتیب احتمال ارسال صفر فریم معادل با e^{-G} است. در فاصلهٔ زمان ارسال دو فریم [یعنی 2t] میانگین فریم تولید شده معادل با 2G است. احتمال آنکه در طول «زمان آسیب‌پذیری» یک فریم، هیچ ترافیک دیگری تولید و ارسال نشود مساوی با $P_0 = e^{-2G}$ است. با اعمال رابطه $S = G \times P_0$ به دست خواهیم آورد:

$$S = G \times e^{-2G}$$

در شکل ۴-۳، رابطهٔ بین ترافیک و بازده مفید کانال (Troughput) نشان داده شده است. بیشترین بازده به ازای $G = 0.5$ بدست می‌آید و در این حالت بازده کانال معادل $S = \frac{1}{2e}$، یعنی چیزی حدود 0.184 خواهد بود. به عبارت دیگر، بیشترین بهرهٔ مورد انتظار کانال (Channel Utilization) چیزی حدود ۱۸ درصد است. این مقدار بهرهٔ کانال چندان جالب نیست ولی در سیستمی که ایستگاه‌ها در زمانِ دلخواه فریم خود را ارسال می‌کنند نمی‌توان انتظار داشت بهرهٔ کانال صددرصد باشد. [یعنی هیچ تصادفی اتفاق نیفتد.]



شکل ۴-۳. بازده مفید کانال برحسب ترافیک عرضه شده در سیستم ALOHA.

Slotted ALOHA

در سال ۱۹۷۲ ، شخصی به نام روبرتز، روشی برای دو برابر کردن ظرفیت مفید سیستم ALOHA ارائه کرد.(مرجع Roberts, 1972) پیشنهاد وی مبنی بر آن بود که زمان به برشهای گسسته‌ای تقسیم شود و هر برش زمان معادل با زمان لازم برای ارسال یک فریم باشد. این روش مستلزم آن بود که کاربران محدودهٔ این برشهای زمانی (Time Slots) را به درستی بدانند. برای رسیدن به چنین هماهنگی می‌توان یک ایستگاه خاص را به خدمت گرفت تا در ابتدای هر برش زمانی سیگنالی همانند سیگنال ساعت منتشر نماید.

در روش روبرتز که امروزه به نام Slotted ALOHA مشهور شده است، برخلاف روش Pure ALOHA هیچ کامپیوتری مجاز نیست وقتی که کلید Enter (Carriage Return) فشار داده شد ، داده‌ها را بر روی کانال بفرستد؛ در عوض باید آنقدر منتظر بماند تا به آغاز برشِ زمان بعدی برسد. بنابراین روش پیوستهٔ Pure ALOHA به روش گسستهٔ Slotted ALOHA تبدیل شده است. از آنجایی که «دورهٔ آسیب‌پذیری» (یعنی زمانی که در خلال آن نباید فریم دیگری ارسال شود) نصف شده است لذا احتمال آنکه هیچ دادهٔ دیگری در خلال یک برش (اسلات) تولید نشود e^{-G} خواهد بود؛ بدین ترتیب بدست می‌آوریم:

$$S = G \times e^{-G}$$ رابطهٔ (۴-۳)

به‌گونه‌ای که از شکل ۴-۳ مشهود است، در سیستم Slotted ALOHA ، بهرهٔ کانال در $G=1$ به مقدار حداکثر خود می‌رسد و در این نقطه بهرهٔ کانال معادل با $S = \frac{1}{e}$ (یعنی حدود 0.368) خواهد بود. اگر این سیستم در شرایط $G = 1$ عمل کند احتمال آنکه یک برش زمانی خالی باشد [و بتوان ارسال موفق داشت] حدود 0.368 خواهد بود. (طبق رابطهٔ ۴-۳) بیشترین موفقیتی که می‌توان از Slotted ALOHA انتظار داشت عبارتست از: ۳۷ درصد برای خالی ماندن یک اسلات، ۳۷ درصد برای ارسال موفق و ۲۶ درصد برای تصادم خواهد بود. اگر این سیستم با مقدارِ بیشترِ G کار کند، تعداد برشهای خالی کاهش یافته و میزان تصادمها به صورت نمائی افزایش خواهد داشت. [G: تلاش برای ارسال G عدد فریم در واحد زمان است و واحد زمان نیز زمان لازم برای ارسال یک فریم می‌باشد.] برای آنکه بررسی کنیم افزایش سریع تعداد تصادمها با افزایش G ، از کجا منشاء می‌گیرد ارسال یک فریم آزمایشی را مَد نظر قرار بدهید: احتمال آنکه این فریم از تصادم جان سالم به در ببرد (یعنی تمام ایستگاههای دیگر در این برش زمانی ساکت باشند) e^{-G} است و بالطبع احتمال تصادم معادل با $1 - e^{-G}$ خواهد بود. احتمال ارسال موفق فریم، منوط به k تلاش پیاپی خواهد بود (یعنی $k-1$ تصادم متوالی و نهایتاً یک ارسال موفق) یعنی:

$$P_K = e^{-G}.(1 - e^{-G})^{k-1}$$

پس از فشار داده شدن کلید Enter ، میانگین دفعات ارسال یعنی E ، معادل است با:

$$E = \sum_{k=1}^{\infty} k.P_k = \sum_{k=1}^{\infty} k.e^{-G}(1 - e^{-G})^{k-1} = e^{-G}$$

در نتیجه، از آنجایی که E به صورت نمائی با G ارتباط دارد اندکی افزایش در بار کانال ، بهرهٔ کانال را بشدت کاهش خواهد داد.

روش Slotted ALOHA بدلائلی که در بدو امر چندان مشهود و روشن به نظر نمی‌رسد، از اهمیت ویژه‌ای برخوردار است. این روش در دههٔ ۱۹۷۰ ابداع گردید، در چند سیستم آزمایشی و ابتدائی به کار گرفته شد و پس از آن تقریباً به دست فراموشی سپرده شد ولیکن وقتی دسترسی به اینترنت از طریق کابل اختراع شد ، ناگاه این مشکل بزرگ به میان آمد که چگونه می‌توان کانالی مشترک را به کاربران رقیب اختصاص داد. اینجا بود که Slotted ALOHA بار دیگر به صحنه آمد. پروتکلهایی بوده‌اند که اگرچه درست و موثر کار می‌کرده‌اند ولی بدلائل سیاسی [غیرعلمی] کنار گذاشته شده‌اند (مثلاً برخی از شرکتهای بزرگ علاقمندند که همه دنباله‌رُوی عملکرد آنها باشند)

ولیکن سالها بعد اشخاص زیرک و با هوش بدین حقیقت می‌رسند که این پروتکلهای فراموش شده می‌توانند مشکلات فعلی آنها را حلّ کنند. به همین دلیل در این فصل به پروتکلهای زیبا و جالبی خواهیم پرداخت که در حال حاضر کاربرد گسترده‌ای ندارند ولیکن ممکن است در کاربردهای آتی مفید واقع شوند؛ بشرط آنکه طراحان شبکه نسبت به آنها آگاهی داشته باشند. البته ما به بررسی پروتکل‌های متعددی نیز پرداخته‌ایم که در حال حاضر کاربرد بسیار گسترده‌ای دارند.

٤ـ۲ـ۲ پروتکلهای دسترسی چندگانه با قابلیت شنود سیگنال حامل (CSMA)

در روش Slotted ALOHA بیشترین بهرهٔ مفیدی که می‌توان بدست آورد 1/e [معادل 0.368] است. این بهره چندان جالب نیست چراکه در این روش هر ایستگاه بدون اعتنا به وضعیت بقیه ایستگاه‌ها و به دلخواه ارسال خود را انجام می‌دهد، لذا در این سیستم تعداد تصادمها زیاد خواهد بود. لیکن در شبکه‌های محلّی امکان آن وجود دارد که هر ایستگاه بتواند تشخیص بدهد دیگر ایستگاه‌ها چه می‌کنند و بر اساس این تشخیص عملکرد خود را تنظیم نماید. در چنین شبکه‌هایی می‌توان به بهرهٔ کانال بسیار بالاتر از 1/e دست یافت. در این بخش چندین پروتکل برای افزایش کارآئی و بهره کانال معرفی می‌کنیم.

پروتکلهایی که در آنها هر ایستگاه به سیگنال حامل روی کانال گوش داده و براساس وضعیت کانال عمل می‌کنند اصطلاحاً «**پروتکلهای شنود حامل**» (Carrier Sense Protocols) نامیده می‌شوند. تاکنون تعداد بی‌شماری از این پروتکلها معرفی شده‌اند. کلینرزاک و توباگی (۱۹۷۵) معدودی از این پروتکلها را تحلیل کرده‌اند. در ذیل چندگونه از پروتکلهای مبتنی بر شنود سیگنال حامل را بررسی خواهیم کرد.

Persistent and Nonpersistent CSMA

اوّلین پروتکل مبتنی بر شنود سیگنال حامل که در اینجا بررسی خواهیم کرد، روش 1-Persistent CSMA است. در این روش هرگاه یک ایستگاه ، داده‌ای برای ارسال داشته باشد ابتدا به کانال گوش می‌دهد تا ببیند آیا در این لحظه کس دیگری در حال ارسال هست یا خیر. اگر کانال مشغول باشد ایستگاه آنقدر منتظر می‌ماند تا کانال آزاد شود؛ ولی اگر ایستگاه ، کانال را آزاد تشخیص بدهد فریم خود را ارسال می‌کند. اگر تصادمی رخ بدهد ایستگاه به اندازهٔ یک زمان تصادفی صبر کرده و تمام مراحل را از نو آغاز می‌نماید. این پروتکل، اصطلاحاً 1-Persistent (پروتکل پافشاری بر ارسال) نامیده می‌شود چراکه وقتی یک ایستگاه کانال را آزاد تشخیص بدهد با احتمال ۱ ارسال خود را آغاز می‌کند. [یعنی اگر ایستگاه کانال را آزاد تشخیص بدهد یقیناً و به صورت غیرمشروط ارسال خود را آغاز می‌نماید. -م]

«**تاخیر انتشار**» (Propagation Delay) تاثیر بسزائی در کارآئی این پروتکل دارد. احتمال ناچیزی وجود دارد که دقیقاً پس از شروع ارسال فریم توسط یک ایستگاه، ایستگاه دیگری نیز آمادهٔ ارسال شده و کانال را بررسی و شنود نماید. اگر سیگنال ایستگاه اوّل هنوز به ایستگاه دوّم نرسیده باشد [بدلیل تاخیر انتشار]، دوّمی نیز کانال را آزاد تشخیص داده و ارسال خود را آغاز می‌کند و طبعاً منجر به تصادم (Collision) خواهد شد. هر چه تاخیر انتشار بیشتر باشد تاثیر مخرب آن بیشتر و منجر به کارآئی بدتر پروتکل خواهد شد.

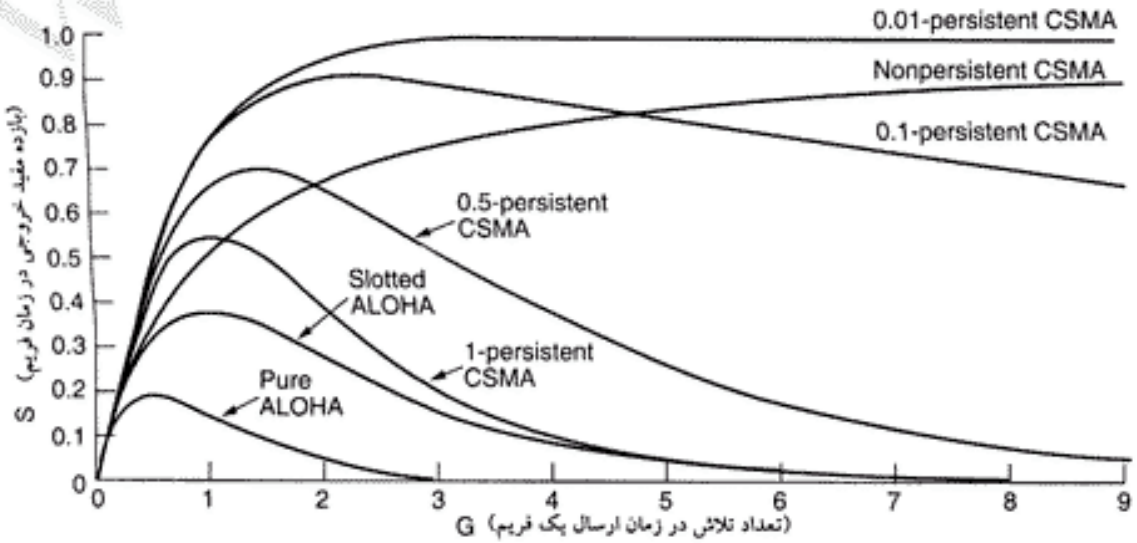
حتی اگر تاخیر انتشار صفر باشد باز هم «تصادم» وجود خواهد داشت: هرگاه دو ایستگاه در خلال ارسال ایستگاه ثالثی آمادهٔ ارسال فریم شوند هر دوی آنها مودبانه منتظر خاتمهٔ ارسال فریم جاری می‌شوند و به محض آزاد شدن کانال بطور همزمان ارسال فریم خود را آغاز می‌نمایند که منجر به تصادم خواهد شد. اگر این دو ایستگاه در ارسال فریم خود عجول و مصّر نبودند تصادم‌های کمتری رخ می‌داد! با این حال این پروتکل بسیار بهتر از Pure ALOHA عمل می‌کند زیرا در این پروتکل دو ایستگاه [با شنود کانال] از تلاقی با ارسال فریم ایستگاه در

حال ارسال اجتناب می‌کنند. می‌توان به صورت ذهنی کارآئی این روش را بسیار بیشتر از Pure ALOHA ارزیابی کرد. بدلیل مشابه، کارآئی این روش از Slotted ALOHA نیز بیشتر است.

دوّمین پروتکل که آن نیز مبتنی بر شنود سیگنال است Nonpersistent CSMA نام دارد. در این پروتکل، تلاش آگاهانه جهت ارسال برروی کانال با اصرار کمتری نسبت به پروتکل قبلی انجام می‌گیرد: هر ایستگاه قبل از ارسال، کانال را بررسی (شنود) می‌کند؛ اگر کسی دیگر در حال ارسال نباشد ایستگاه، فریم خودش را می‌فرستد ولیکن هرگاه کانال از قبل در اختیار دیگری باشد، ایستگاه بطور دائم به شنود کانال **نخواهد پرداخت**. در عوض [هرگاه ایستگاه کانال را مشغول تشخیص بدهد] به اندازه یک زمان تصادفی صبر کرده و پس از آن الگوریتم فوق را تکرار می‌کند. [تنها تفاوت این روش با روش قبلی در آنست که هرگاه کانال مشغول باشد ایستگاه مترصد آزاد شدن آن باقی نمی‌ماند و به اندازه یک زمان تصادفی کانال را به حال خود رها می‌کند و بدان گوش نمی‌دهد. -م] این الگوریتم طبعاً بهرهٔ کانال بهتری را ارائه می‌دهد [چرا که تصادمها در لحظهٔ آزاد شدن کانال کاهش می‌یابد] ولیکن تاخیر بیشتری نسبت به روش 1-Persistent CSMA دارد. [تاخیر ارسال]

آخرین پروتکل، p-Persistent CSMA نام دارد. این روش فقط برروی کانالهای زمان‌بندی‌شده (Slotted Time Channels) قابل اعمال است و بدین ترتیب عمل می‌کند: هرگاه ایستگاهی آمادهٔ ارسال شود ابتدا کانال را شنود می‌نماید؛ اگر کانال آزاد باشد فریم خود را با احتمال p ارسال می‌کند و یا به احتمال $q=1-p$ ارسال خود را تا فرا رسیدن برش بعدی زمان [اسلات بعدی] به تعویق می‌اندازد. یعنی حتی اگر یک اسلات خالی باشد ممکن است با احتمال p فریم خود را بفرستد یا با احتمال q به تعویق بیندازد. این فرآیند آنقدر تکرار می‌شود تا آنکه یا فریم ارسال شود یا آنکه ایستگاهی دیگر ارسال خود را آغاز نماید. در حالت دوّم [یعنی ایستگاهی دیگر موفق به ارسال شود] ایستگاه ناموفق، همانند وقتی که تصادم رخ داده عمل می‌کند یعنی به اندازهٔ یک زمان تصادفی صبر کرده و از نو شروع می‌نماید. اگر ایستگاه در همان ابتدا کانال را مشغول تشخیص بدهد تا اسلات بعدی صبر می‌کند و الگوریتم فوق را به اجرا می‌گذارد.

شکل ۴-۴ منحنی ظرفیت مفید (Throughput) کانال را بر مبنای حجم ترافیک تولید شده، برای سه پروتکل فوق و پروتکلهای Pure ALOHA و Slotted ALOHA نشان می‌دهد.

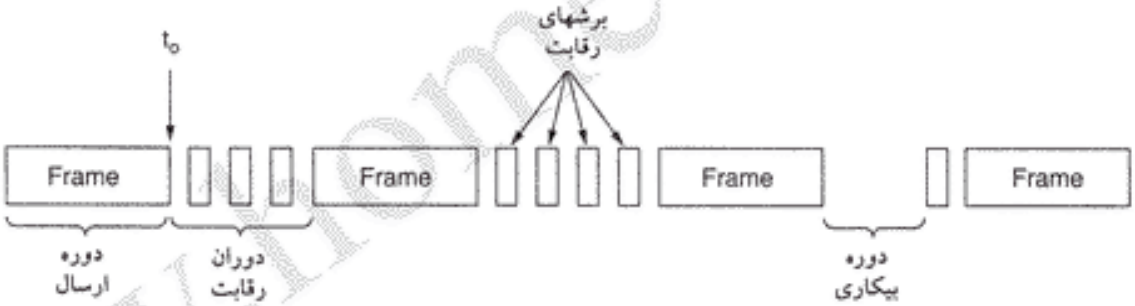


شکل ۴-۴. مقایسهٔ بهره‌وری کانال (ظرفیت مفید) بر حسب بار برای پروتکلهای گوناگون دسترسی تصادفی به کانال.

پروتکل CSMA با تشخیص تصادم

پروتکلهای Persistent & Nonpersistent CSMA به روشنی بهینه‌تر از ALOHA هستند زیرا مطمئناً اگر ایستگاهی کانال را مشغول تشخیص بدهد ارسال خود را آغاز نخواهد کرد. بهبود دیگر این روشها آنست که ایستگاهها به محض آنکه از وقوع تصادم آگاه شدند ارسال خود را نیمه کاره رها کنند. به عبارت دیگر هرگاه دو ایستگاه، کانال را آزاد احساس کرده و همزمان شروع به ارسال نمایند، تقریباً هر دوی آنها بلافاصله از وقوع تصادم مطلع خواهند شد. در چنین حالتی به محض کشف پدیده تصادم، ایستگاهها بجای ارسال کامل فریمهای آسیب دیده، به ارسال خود خاتمه می‌دهند. قطع سریع ارسال فریمهای آسیب‌دیده ، در زمان و پهنای باند صرفه‌جویی خواهد کرد. چنین پروتکلی که اصطلاحاً CSMA/CD[1] نام دارد بطور گسترده در شبکه‌های محلی به کار گرفته شده است. به ویژه، این پروتکل مبنای شبکهٔ محلی و شناخته شده **اترنت** است، لذا ارزش آنرا دارد که اندک زمان بیشتری برای بررسی جزئیات آن صرف کنیم.

CSMA/CD نظیر بسیاری از پروتکلهای دیگر LAN از مدل مفهومی نشان داده شده در شکل ۴-۵ تبعیت می‌کند. در لحظه‌ای که با نماد t_0 مشخص شده، ایستگاهی ارسال فریم خود را به پایان رسانده است. در این لحظه ایستگاههایی که فریمی برای ارسال دارند ممکن است برای ارسال آن تلاش کنند. اگر دو یا چند ایستگاه بطور همزمان تصمیم به ارسال بگیرند تصادم رخ خواهد داد. وقوع تصادم را می‌توان با بررسی توان مصرفی یا اندازه‌گیری و مقایسهٔ پهنای پالس سیگنال دریافتی از کانال و مقایسهٔ آن با سیگنال ارسالی تشخیص داد.



شکل ۴-۵ CSMA/CD می‌تواند در یکی از سه وضعیت: «رقابت»، «ارسال» یا «بیکار» قرار داشته باشد.

پس از آنکه یک ایستگاه متوجه وقوع تصادم شد، ارسال خود را ناتمام رها کرده و از نو شروع می‌کند؛ (البته با این فرض که ایستگاه دیگری در این بینابین ارسال خود را آغاز ننماید.) بدین ترتیب مدل ارائه شده برای CSMA/CD ، شامل: (۱) چندین مرحلهٔ متناوب «**رقابت**» (Contention) (۲) بازه‌های ارسال و (۳) بازه‌های بیکاری خواهد بود. (بازه‌های بیکاری بدین معناست که تمام ایستگاه‌ها بدلیل عدم نیاز به ارسال ساکت بوده‌اند.)

حال جزئیات الگوریتم رقابت را دقیقتر بررسی می‌نمائیم: فرض کنید دو ایستگاه بطور همزمان ارسال فریم خود را در زمان t_0 آغاز نمایند. چقدر طول می‌کشد تا متوجه شوند تصادم اتفاق افتاده است؟ پاسخ این سوال برای تعیین طول زمان رقابت و همچنین محاسبهٔ تاخیر و ظرفیت مفید کانال حیاتی است.

حداقل زمان لازم برای تشخیص وقوع تصادم، معادل زمانی است که طول می‌کشد تا سیگنال ارسالی از یک ایستگاه به ایستگاه دیگر منتشر شود. براساس این استدلال شاید شما تصور کنید که یک ایستگاه تا پس از زمانی معادل با زمان انتشار سیگنال برروی کلّ کابل از وقوع تصادم مطلع نشده و از تصرف کانال، مطمئن نخواهد بود. (منظورمان از «**تصرّف**» کانال آنست که ایستگاه‌های دیگر متوجه شوند که او در حال ارسال است و تداخلی پیش نیاید.) این نتیجه‌گیری صحیح نیست. به سناریوی زیر که در بدترین شرایط در نظر گرفته شده، دقت نمایید: زمان

---

۱. Carrier Sense Multiple Access with Collision Detection

لازم برای انتشار سیگنال بین دورترین ایستگاه‌ها از یکدیگر را τ فرض کرده‌ایم؛ در لحظهٔ t_0 یک ایستگاه ارسال خود را آغاز می‌کند. در لحظهٔ $\tau - \varepsilon$ یعنی دقیقاً قبل از لحظه‌ای که سیگنال به دورترین ایستگاه می‌رسد، آن ایستگاه شروع به ارسال می‌کند. مشخصاً این ایستگاه وقوع تصادم را کشف می‌کند ولیکن سیگنال نویزی که در اثر تصادم تولید می‌شود تا زمان $2\tau-\varepsilon$ به ایستگاه مبداء باز نخواهد گشت. به عبارت دیگر در بدترین حالت، تا انقضای زمان 2τ پس از شروع ارسال ، ایستگاه مبداء نمی‌تواند از تصرف کانال مطمئن باشد. بهمین دلیل ما بازهٔ رقابت را همانند روش Slotted ALOHA مدل کرده‌ایم که در آن پهنای هر برش زمانی (اسلات) معادل 2τ می‌باشد. بر روی یک کابل کواکسیال به طول ۱ کیلومتر، 2τ حدوداً معادل ۵ میکروثانیه است. برای سادگی فرض را بر آن خواهیم گذاشت که هر برش زمان در برگیرنده تنها یک بیت [به طول 2τ] است و به محض آنکه کانال تصرف شد، ایستگاه می‌تواند با هر سرعت دلخواه ارسال خود را انجام بدهد ولیکن بدیهی است که با ارسال با نرخ $1bit/2\tau$ نخواهد بود!

درک این موضوع که فرآیند کشف تصادم به صورت آنالوگ انجام می‌شود اهمیت دارد. سخت‌افزار هر ایستگاه باید در حین ارسال به کابل گوش بدهد. اگر آنچه را که از کابل بازخوانی می‌کند با آنچه برروی کابل قرار داده، متفاوت باشد متوجه وقوع تصادم می‌شود. کشف تصادم مستلزم آنست که روش کدینگ سیگنال این امکان را فراهم بیاورد (زیرا مثلاً تصادم دو سیگنال صفر ولت قابل کشف نیست). به همین دلیل معمولاً از روشهای کدینگ خاصّ [مثل منچستر] استفاده می‌شود.

بدیهی است که ایستگاه فرستنده باید بطور مدام بر کانال نظارت کند و منتظر شنیدن سیگنال نویزی که مشخص کنندهٔ وقوع تصادم است باقی بماند. بهمین دلیل CSMA/CD با یک کانال منفرد ذاتاً سیستمی Half Duplex (دو طرفهٔ غیرهمزمان) است. در چنین سیستمی برای ایستگاه این امکان وجود ندارد که ارسال و دریافت فریمها را بطور همزمان انجام بدهد زیرا بخش گیرندهٔ آن حتی در خلال ارسال مشغول و در حال پیگیری بروز تصادم برروی کانال است. [البته در حین ارسال، گیرنده فقط آمادهٔ کشف تصادم به صورت آنالوگ است و داده‌های روی کانال، دریافت یا ذخیره نمی‌شوند.]

لازم است بدین نکتهٔ بدیهی اشاره کنیم که **پروتکل زیرلایهٔ MAC دریافت مطمئن فریمها را تضمین نمی‌کند** چراکه اگر حتی تصادمی رخ ندهد گیرندهٔ فریم ممکن است بدلائل گوناگون نتواند فریم را به درستی دریافت کند. (دلائلی مثل فقدان فضای بافر یا وقفه‌های گمشده)

۳-۲-٤ پروتکلهای بدون تصادم

اگرچه در روش CSMA/CD پس از آنکه یک ایستگاه بطور مطمئن کانال را تصرف کرد دیگر وقوع تصادم منتفی است ولیکن وقوع تصادمهای مکرّر در دورهٔ رقابت[1] کاملاً طبیعی است. این تصادمها تاثیر زیانباری برروی کارآئی سیستم خواهد داشت؛ بالاخص زمانی که کابل، طولانی (و در نتیجه τ یا همان تاخیر انتشار بالاست) و یا فریمها کوتاه هستند، این تاثیر بسیار مخرّب است. در چنین حالاتی روش CSMA/CD قابل‌استفاده نخواهد بود. در این بخش چند پروتکل را بررسی می‌کنیم که در آنها مسئله رقابت حَل شده و بروز تصادم حتی در زمان رقابت منتفی است. اگرچه بسیاری از این روشها در حال حاضر برروی سیستمهای شناخته شده به کارگرفته نشده است ولیکن در اختیار داشتن پروتکلهایی با ویژگیهای عالی، برای به کارگیری در سیستمهای پیشرفتهٔ آینده یک حُسن محسوب می‌شود!

در پروتکلهایی که بررسی می‌شوند فرض را بر آن گذاشته‌ایم که دقیقاً N ایستگاه به کانال مشترک متصلند و

---

۱. Contention Period

هر ایستگاه آدرسی بین 0 تا N−1 دارد که درون سخت‌افزار ایستگاه حکّ شده است. این موضوع که ممکن است برخی از ایستگاه‌ها در برخی از مواقع غیر فعال و خاموش باشند چندان اهمیت ندارد. همچنین فرض کرده‌ایم که تاخیر انتشار قابل صرفنظر باشد. مسئلهٔ بنیادی هنوز باقی است: «پس از آنکه ارسال فعلی یک ایستگاه به پایان رسید کدامین ایستگاه حقّ دارد کانال را در اختیار بگیرد و فریم خود را ارسال نماید؟»

در این بخش نیز از مدل شکل ۴-۵ با برشهای گسستهٔ رقابت[1] استفاده می‌کنیم.

یک پروتکل مبتنی بر نشانه‌های بیتی (Bit Map)

در اولین پروتکل بدون تصادم که Basic Bit-Map Method نام دارد بازهٔ رقابت متشکل از دقیقاً N برش زمانی است. هرگاه ایستگاه شمارهٔ صفر، فریمی آمادهٔ ارسال داشته باشد در برش (اسلات) شماره صفر، بیت ۱ را برروی کانال می‌گذارد؛ هیچ ایستگاه دیگری حق ندارد در این برش چیزی برروی کانال بگذارد. فارغ از آنکه ایستگاه صفر چه کاری می‌کند ایستگاه شماره ۱ نیز این فرصت را دارد که در برش شماره ۱ با قرار دادن بیت ۱ برروی کانال تقاضای ارسال فریم بدهد. (البته اگر فریمی برای ارسال آماده داشته باشد.) بطور کلّی ایستگاه شماره j حق دارد در صورتی که فریمی جهت ارسال داشته باشد در برش شمارهٔ j با گذاشتن بیت ۱ برروی کانال، تقاضای خود را اعلام نماید. پس از آنکه کلّ N برش رقابت سپری شد تمام ایستگاه‌ها فهرست کاملی از ایستگاه‌های متقاضی ارسال ، در اختیار دارند. پس از این لحظه، ایستگاههای متقاضی ارسال، به ترتیب شماره، ارسال فریمهای خود را آغاز می‌کنند. (شکل ۴-۶ را ملاحظه کنید)

شکل ۴-۶. پروتکل پایه مبتنی بر نشانه‌های بیتی (Basic Bitmap Protocol).

از آنجایی که تمام ایستگاه‌ها در هر لحظه می‌دانند که چه ایستگاهی حقّ ارسال دارد به هیچ وجه تصادم رخ نخواهد داد. پس از آنکه آخرین ایستگاه، فریم خود را ارسال کرد، (رخدادی که همه ایستگاهها بسادگی می‌توانند متوجه آن شوند) مجدداً دورهٔ رقابت، شامل N برش (یا بعبارتی N بیت) شروع می‌شود. هرگاه ایستگاهی پس از گذشت برش (اسلات) متعلق به او ، آمادهٔ ارسال شود فرصت را از دست داده و بایستی خاموش بماند تا دیگر ایستگاه‌ها شانس خود را آزموده و کارشان را انجام بدهند تا دور بعدی رقابت فرا برسد. پروتکلهایی همانند این روش که در آنها هر ایستگاه تقاضای خود را قبل از ارسال و به صورت فراگیر به اطلاع همه می‌رساند اصطلاحاً پروتکل‌های رزروسازی (Reservation) نامیده می‌شوند.

اجازه بدهید مختصراً به کارآئی این پروتکل بپردازیم. برای سادگی ، واحد زمان را معادل طول بیت هر برش رقابت در نظر گرفته‌ایم و طول هر فریم داده را معادل d واحد زمان فرض کرده‌ایم. [یعنی هر یک از برشهای رقابت را یک واحد و بر این اساس، طول فریم را d واحد زمان، تلقی کرده‌ایم.] در شرایطی که بار ایستگاهها پائین است، برشهای رقابت بطور متوالی تکرار می‌شوند و بدلیل عدم تقاضا برای ارسال فریم مکرراً خالی می‌مانند. حال وضعیت را از دیدگاه یک ایستگاه با شماره پائینی مثل صفر یا یک بررسی می‌کنیم. [بدون آنکه به عمومیت استدلال لطمه‌ای بخورد] وقتی چنین ایستگاهی آماده ارسال می‌شود باید تا رسیدن برش متناظر با شمارهٔ خودش صبر کند

---

۱. Discrete Contention slots

در حالیکه ممکن است شمارهٔ برش فعلی یکی از شماره‌های میانی باشد. بطور «میانگین» ایستگاه مجبور است به اندازهٔ N/2 صبر کند تا دور فعلی خاتمه یافته و در برش متعلق به خودش تقاضا بدهد؛ سپس باید اندازهٔ N برش صبر کند تا تمام برشها به ترتیب بگذرند و او بتواند ارسال خود را آغاز کند. [چون نوبت او گذشته بطور میانگین باید N/2 برش دیگر صبر کند تا زمان ارسال او فرا برسد. بنابراین بطور میانگین زمانی معادل 1.5N معطل می‌شود.]

چشم‌انداز شانس ایستگاه‌های با شمارهٔ بالا روشنتر است. عموماً چنین ایستگاه‌هایی بطور میانگین مجبورند قبل از شروع به ارسال، N/2 برش صبر کند تا نوبت به اعلام تقاضا و سپس ارسال آنها برسد. ایستگاه‌های با شمارهٔ بالا به ندرت مجبورند که یک دور کامل صبر کنند تا دور بعدی فرا برسد. از آنجایی که ایستگاه‌های با شمارهٔ پائین باید بطور میانگین 1.5N و ایستگاه‌های با شمارهٔ بالا باید 0.5 N منتظر بمانند لذا میانگین این دو زمان N خواهد شد. با این استدلال، محاسبهٔ کارآئی کانال در بار پائین ساده است: میزان سرباری که باید برای هر فریم d بیتی متحمل شد N بیت است بنابراین این کارآئی کانال در بار پایین عبارتست از: $d/(N + d)$

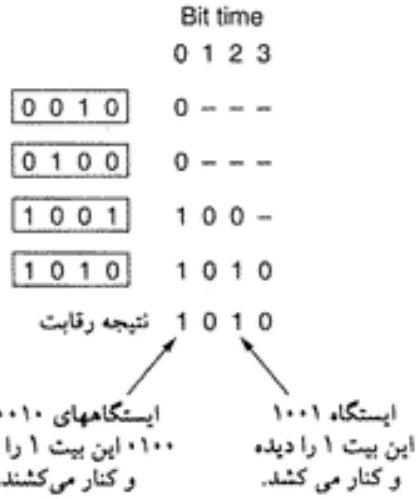
در بار بالا، یعنی وقتی تمام ایستگاه‌ها چیزی برای ارسال داشته باشند، N برش رقابت برای ارسال N تا فریم متوالی صرف می‌شود و این سربار به ازای هر فریم ۱ بیت خواهد بود [یعنی N بیت سربار برروی N فریم d بیتی سرشکن می‌شود لذا به ازای هر فریم d بیتی فقط یک بیت سربار تحمیل می‌شود.] بنابراین کارآئی کانال در بار بالا عبارتست از: $d/(d+1)$. میانگین تاخیر ارسال هر فریم نیز، معادل با زمان تاخیر در صفّ داخلی هر ایستگاه به اضافهٔ مقدار $N.(d + 1)/2$ (معادل تاخیر ارسال پس از آنکه فریم به سر صف داخلی ایستگاه می‌رسد) خواهد بود.

روش شمارش دودوئی معکوس (Binary Countdown)

مشکل پروتکل Bitmap آنست که به ازای هر ایستگاه یک بیت سربار تحمیل می‌شود فلذا شبکه قابل‌گسترش به مقیاس هزاران ایستگاه، نخواهد بود. می‌توان براساس آدرس دودوئی ایستگاه‌ها، فرآیند رزروسازی را به روش بهتری انجام داد: ایستگاهی که می‌خواهد کانال را به خدمت بگیرد شماره آدرس خود را به صورت دنباله‌ای از بیتها که از پرارزشترین آن شروع می‌شود برروی کانال منتشر می‌کند. فرض بر آنست که تمام آدرسها دارای طول یکسانی هستند. هر یک بیتهای آدرس پس از انتشار برروی کانال بطور منطقی با یکدیگر OR (Wired OR) می‌شوند. این روش که «**پروتکل شمارش دودوئی معکوس**» نام دارد برای اولین بار در سیستم Datakit (Fraser,1987) به کارگرفته شد. در این روش به صراحت فرض شده که تاخیر انتشار خط ناچیز است و تمام ایستگاه‌ها می‌توانند بطور همزمان بیتهای ارسالی را بر روی کانال را بشنوند.

برای پیشگیری از هرگونه برخورد، باید یک «قاعدهٔ داوری» (Arbitration Rule) اعمال شود: به محض آنکه ایستگاهی متوجه شود که بیت پرارزش صفر او که برروی کانال قرار گرفته با بیت ۱ بازنویسی شده از دور رقابت کنار می‌کشد. به عنوان مثال اگر ایستگاه‌های 0010 ، 0100 ، 1001 و 1010 بخواهند کانال را بدست بیاورند، همگی در اولین برش زمانی، بیت پرارزش خود را برروی کانال می‌گذارند، یعنی به ترتیب بیتهای 0 ، 0 ، 1 و 1 برروی کانال ظاهر می‌شود. این بیتها با یکدیگر Wired OR شده و نتیجهٔ آن ۱ خواهد بود. ایستگاه‌های 0010 و 0100 متوجه ظهور ۱ برروی خط شده و نتیجه می‌گیرند که ایستگاهی با شمارهٔ بالاتر برای تصرّف کانال رقابت می‌کند، لذا از دور فعلی خارج می‌شوند. در ادامه فقط ایستگاه‌های 1001 و 1010 به رقابت می‌پردازند.

بیت بعدی هر دو صفر است فلذا این دو ایستگاه بازهم ادامه می‌دهند. بیت بعدی روی کانال ۱ خواهد بود لذا ایستگاه 1001 کنار می‌کشد. در نهایت ایستگاه 1010 برندهٔ این رقابت خواهد بود چراکه دارای بالاترین شمارهٔ آدرس است. پس از پیروزی در این رقابت (که به صورت مزایده برگزار شده)، ایستگاه برنده می‌تواند فریم خود را ارسال کند و پس از آن دور بعدی «مزایده» آغاز می‌شود. عملکرد این پروتکل در شکل ۴-۷ به تصویر کشیده شده



شکل ۴ـ۷. پروتکل شمارش دودویی معکوس. خط تیره علامت سکوت ایستگاه است.

است. پروتکل فوق دارای این ویژگی است که ایستگاه‌های با شماره بالاتر اولویت بیشتری نسبت به ایستگاه‌های با شمارهٔ پائینتر دارند؛ این ویژگی براساس زمینه و نوع کار می‌تواند خوب یا بد باشد.

کارآئی کانال در این روش معادل $d/(d + \log_2 N)$ است. با اینحال اگر قالب فریم بگونه‌ای زیرکانه انتخاب شود که آدرس فرستندهٔ فریم، همان اولین فیلد باشد مقدار سربار $\log_2 N$ بیت نیز تلف نخواهد شد و کارآئی کانال صددرصد است!

دو پژوهشگر به نامهای «مارک» و «وارد» (۱۹۷۹)، گونه‌ای از روش «شمارش دودوئی معکوس» را با استفاده از واسطهای موازی (Parallel Interfaces) به جای واسطهای سریال معرفی و تشریح کردند. [یعنی ارسال اطلاعات به جای سریال به صورت موازی انجام می‌شود.] آنها پیشنهاد کردند که برای آدرس‌دهی ایستگاه‌ها، [به جای شماره‌های ثابت] از شماره‌های مجازی استفاده شود. پس از آنکه ایستگاهی موفق به ارسال شد، به شمارهٔ تمام ایستگاه‌های قبل از آن، تا شمارهٔ صفر، یک واحد اضافه می‌شود تا در مرحلهٔ بعد، آنهایی که موفق به ارسال نشده‌اند اولویت بالاتری داشته باشند. [بدین ترتیب مسئله قبضه شدن کانال توسط یک ایستگاه حلّ خواهد شد.-م]

بعنوان مثال فرض کنید ایستگاه‌های C، H، D، A، G، B، E و F به ترتیب دارای اولویتهای ۷، ۶، ۵، ۴، ۳، ۲، ۱ و ۰ هستند و [از بین ایستگاه‌های متقاضی مثل D، A، B و F]، D موفق به ارسال فریم خود شده است. پس از ارسال، D به انتهای این فهرست رفته و ترتیب اولویتها به صورت C، H، A، G، B، E، F و D تغییر می‌کند. بدین ترتیب C به صورت مجازی ایستگاه شماره ۷ باقی می‌ماند ولی شمارهٔ A از ۴ به ۵ صعود و شماره D، از ۵ به صفر سقوط می‌کند. حال ایستگاه D فقط در صورتی قادر به در اختیار گرفتن کانال خواهد شد که هیچ ایستگاه دیگری بدان احتیاج نداشته باشد.

روش شمارش دودوئی معکوس نمونه‌ای است از یک پروتکل ساده، جالب و کارآمد که در انتظار کشف مجدّد، روزگار می‌گذراند و در آینده، منزلتی را برای خود خواهد یافت!!

٤ـ۲ـ٤ پروتکلهای با رقابت محدود

تاکنون دو استراتژی بنیادی را برای دستیابی به کانال در شبکه‌های مبتنی بر کابل، بررسی کرده‌ایم: روشهای رقابت و تصادم مثل CSMA و روشهای بدون تصادم. هر کدام از این استراتژیها را می‌توان بر اساس دو معیار مهّم رده‌بندی کرد: (۱) میزان تاخیر در بار پائین (۲) کارآئی و بهرهٔ کانال در بار بالا.

در بار پایین روشهای مبتنی بر رقابت (مثل ALOHA یا CSMA) ارجحتر است چراکه تاخیر ناچیزی دارند؛ ولیکن وقتی بار افزایش می‌یابد روشهای مبتنی بر رقابت ارزش خود را از دست می‌دهند زیرا سربار تحمیل شده در اثر مبارزه بر سر بدست آوردن کانال، به شدت افزایش می‌یابد. دقیقاً عکس این موضوع در مورد پروتکلهای بدون تصادم صادق است: در بار پائین تاخیر بالائی دارند ولی وقتی بار شبکه افزایش می‌یابد (برعکس پروتکلهای مبتنی بر رقابت)، کارآئی کانال رو به افزایش می‌گذارد.

روشن است که اگر بتوانیم دو ویژگی ممتاز این روشها را با هم ترکیب کنیم به یک پروتکل مطلوب خواهیم رسید که در بار پائین بروش رقابت عمل می‌کند (تا تاخیر کمی داشته باشد) ولی در بار بالا از روش بدون تصادم بهره می‌گیرد (تا کارآئی کانال افزایش یابد). این پروتکلها به نام «**پروتکلهای با رقابت محدود**» مشهور هستند و بررسی آنها، مطالعات ما را در مورد شبکه‌های مبتنی بر شنود سیگنال، به نتیجه نهائی می‌رساند.

پروتکلهای مبتنی بر رقابت که تاکنون بررسی کرده‌ایم «متقارن» (Symmetric) هستند بدین معنا که هر کدام از ایستگاه‌ها، با احتمال p تلاش می‌کند کانال را در اختیار بگیرد و برای تمام ایستگاه‌ها، این احتمال یکسان و مساوی p فرض شده است. می‌توان کارآئی کلّ سیستم را با انتساب احتمالات مختلف به ایستگاه‌های متفاوت، بهبود داد.

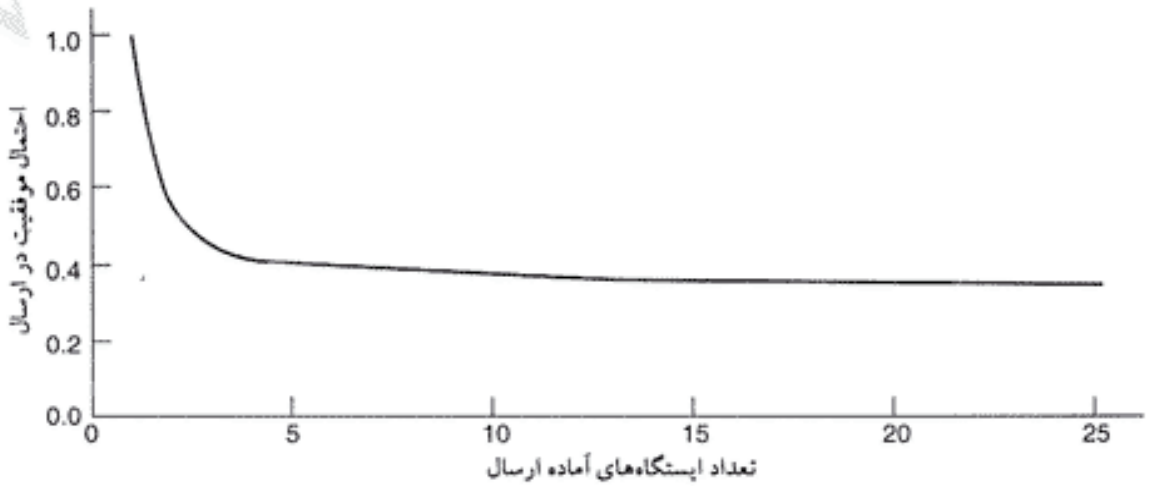
قبل از آنکه به پروتکلهای نامتقارن (Asymmtric) بپردازیم اجازه بدهید تا مروری اجمالی بر کارآئی شبکه در حالت متقارن داشته باشیم. فرض کنید تعداد k ایستگاه برای دسترسی به کانال رقابت می‌کنند و هر کدام در ابتدای یک برش زمان، با احتمال p اقدام به ارسال می‌نمایند. احتمال آنکه یکی از ایستگاه‌ها موفق به در اختیار گرفتن کانال در یکی از برشهای زمانی شود عبارتست از:

$$k.p.(1-p)^{k-1}$$

به منظور پیدا کردن مقداری بهینه برای p [به نحوی که احتمال فوق به حداکثر برسد] از رابطهٔ فوق بر حسب p مشتق گرفته و حاصل را مساوی صفر قرار می‌دهیم؛ سپس p را محاسبه می‌نمائیم. پس از محاسبه، بهترین مقدار p معادل 1/k بدست می‌آید. با قرار دادن 1/k به جای p در رابطهٔ فوق خواهیم داشت:

$$P_s[\text{موفقیت در ارسال با بالاترین احتمال}] = [\frac{k-1}{k}]^{k-1} \qquad \text{رابطه (۴-۴)}$$

منحنی این رابطه در شکل ۴-۸ ترسیم شده است. اگر تعداد ایستگاه‌ها کم باشد، احتمال موفقیت بالاست ولی به محض آنکه تعداد ایستگاه‌ها به ۵ یا بیشتر می‌رسد، احتمال موفقیت در ارسال، به مقدار تقریباً ثابت 1/e ، میل می‌کند.



شکل ۴-۸ منحنی احتمال موفقیت در تصرف کانال برای یک کانال متقارن.

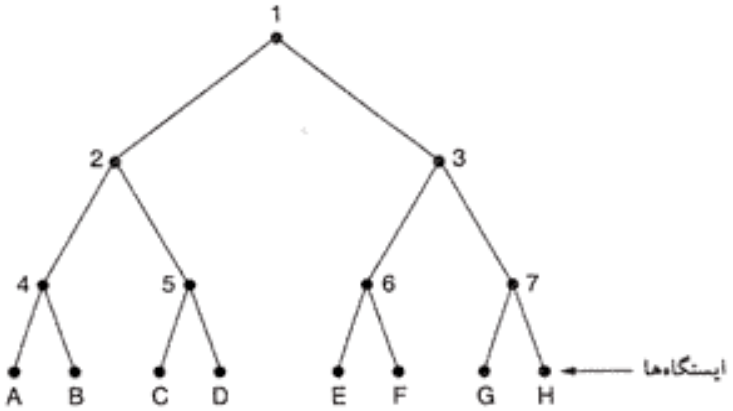
از شکل ۴-۸ بخوبی مشهود است که می‌توان احتمال موفقیت ایستگاه‌ها را با کاهش دادن حجم رقابت، افزایش داد؛ «پروتکلهای با رقابت محدود» سعی می‌کنند دقیقاً همین کار را انجام بدهند. این پروتکلها ابتدا ایستگاه‌ها را به چند گروه تقسیم می‌نمایند. (لزومی به از هم جدا بودن گروه‌ها نیست.) در برش زمانی شمارهٔ صفر، فقط اعضای گروه شمارهٔ صفر حق شرکت در رقابت را دارند. اگر یکی از آنها موفق شد، کانال را در اختیار می‌گیرد و فریم خود را ارسال می‌نماید. اگر برش زمانی شمارهٔ صفر خالی ماند یا تصادم رخ داد، اعضای گروه شمارهٔ ۱ در برش شمارهٔ ۱ رقابت می‌کنند و کار به همین ترتیب ادامه می‌یابد. با تقسیم‌بندی صحیح و مناسب ایستگاه‌ها به چند گروه، میزان رقابت در هر برش زمانی کاهش یافته و طبق منحنی شکل ۴-۸، شبکه در نزدیکی سمت چپ نمودار کار می‌کند.

نکته ظریف در اینجاست که ایستگاه‌ها را چگونه به برشهای زمانی مستقل منتسب نمائیم. [یعنی تعیین آنکه چه ایستگاه‌هایی در چه برش زمانی رقابت کنند. -م] یکی از حالات ویژه آنست که هر گروه صرفاً یک عضو بیشتر نداشته باشد. چنین انتسابی تضمین کنندهٔ آنست که هیچ تصادمی بوجود نخواهد آمد چرا که برای تصاحب هر برش زمانی فقط یک ایستگاه رقابت می‌کند. با چنین پروتکلهایی قبلاً برخورد کرده‌ایم (مثل روش شمارش دودوئی معکوس). حالت خاص دیگر آنست که در هر گروه دو ایستگاه قرار بگیرد. احتمال آنکه هر دوی آنها بطور همزمان و در یک برش زمانی بخواهند ارسال داشته باشند معادل p^2 است که برای مقادیر کوچک p، بسیار ناچیز خواهد بود. هر چه تعداد ایستگاه‌ها در هر گروه بیشتر باشد احتمال تصادم در برشهای زمانی متعلق به آن گروه افزایش خواهد یافت ولی در عوض تعداد برشهای زمانی لازم برای رقابت گروه‌ها کاهش می‌یابد. در محدودترین حالت تمام ایستگاه‌ها در یک گروه واحد قرار می‌گیرند؛ در این حالت عملکرد آن همانند روش Slotted ALOHA خواهد بود. بالطبع به روشی نیاز داریم که برشهای زمانی (Time Slots) را بطور پویا به ایستگاه‌ها اختصاص بدهد یعنی وقتی بار کم است تعداد زیادی ایستگاه در یک برش رقابت کنند ولیکن وقتی بار بالاست تعداد کمی ایستگاه (حتی یک ایستگاه) در هر گروه رقابت نمایند.

پروتکل پیمایش وفقی درخت (Adaptive Tree Walk)

یکی از روشهای بسیار ساده برای انجام عمل انتساب فوق‌الذکر، الگوریتمی است که توسط ارتش ایالات متحده در جنگ جهانی دوم برای آزمایش سربازان و تشخیص بیماری سیفلیس ابداع شد. (Dorfman,1943) ارتش، نمونهٔ خون N سرباز را می‌گرفت و بخشی از این نمونه‌های خون، درون یک لولهٔ آزمایش واحد ریخته و با هم مخلوط می‌شد. سپس این نمونهٔ مخلوط شده، مورد آزمایش قرار می‌گرفت و اگر هیچ آنتی‌بادی، در آن پیدا نمی‌شد تمام سربازان سالم تشخیص داده می‌شدند. در صورت تشخیص آلودگی در نمونهٔ خون، دو نمونهٔ دیگر از خونها تهیه می‌شد: نمونهٔ اوّل شامل مخلوطی از نمونهٔ خون سربازان 1 تا N/2 و نمونهٔ دومی از خون مابقی افراد. این فرآیند آنقدر تکرار می‌شد تا سربازان آلوده مشخص شوند.

برای نسخهٔ کامپیوتری این الگوریتم (Capetanakis,1979)، ساده‌تر آنست که ایستگاه‌ها را به عنوان برگهای یک درخت دودوئی فرض کنید (همانند آنچه که در شکل ۴-۹ می‌بینید). در اولین برش رقابت، (یعنی پس از خاتمهٔ ارسال یک ایستگاه و آزاد شدن کانال) تمام ایستگاه‌ها مجازند برای در اختیار گرفتن کانال رقابت کنند. اگر تصادمی به وقوع پیوست، در برش شمارهٔ ۱، فقط ایستگاه‌هایی که در پوشش گره ۲ از درخت واقعند به رقابت می‌پردازند. اگر یکی از آنها کانال را تصاحب و فریم خود را ارسال کرد، برش زمان بعدی، برای رقابت ایستگاه‌های تحت پوشش گره ۳ در نظر گرفته می‌شود ولیکن اگر بیش از یکی از ایستگاههای تحت پوشش گره ۲، بخواهند ارسال داشته باشند در همان برش شمارهٔ ۱ تصادم رخ می‌دهد و در برش شمارهٔ ۲ نوبت به رقابت ایستگاه‌های گره ۴ خواهد بود.



شکل ۴ـ۹. یک درخت برای هشت ایستگاه.

هرگاه در برش شمارهٔ ۰ تصادمی رخ بدهد، کلّ درخت به صورت «عمقی»[1] (Depth First) پیمایش می‌شود تا ایستگاه‌ها به ترتیب شناسایی و تعیین موقعیت شده و ارسال خود را انجام بدهند. هر برش رقابت، به یک گره خاص در درخت تعلق دارد. اگر تصادمها تکرار شود، پیمایش و جستجو از چپ به راست و به صورت بازگشتی (Recursive) ادامه می‌یابد. اگر یکی از برشها خالی بماند یا فقط یک ایستگاه، بدون تصادم کانال را صاحب شود جستجو در آن گره خاتمه می‌یابد، چراکه تمام ایستگاه‌های آمادهٔ ارسال در آن گره، مشخص شده‌اند. (اگر بیش از یک ایستگاه در آن گره تمایل به ارسال می‌داشتند تصادم رخ می‌داد.)

وقتی بار شبکه بالا است به ندرت اتفاق می‌افتد که مسئله تخصیص کانال در همان برش رقابت شمارهٔ صفر که متعلق به گره ۱ است حلّ شود زیرا به احتمال زیاد بیش از یک ایستگاه آمادهٔ ارسال هستند. بدلیل مشابه این مسئله در برش ۲ و ۳ نیز حل نخواهد شد و رقابت به مراحل بعدی خواهد کشید. سوال آنست که بطور کلّی، جستجو از چه سطحی آغاز شود؟ روشن است که هر چه بار سنگینتر باشد، جستجو باید از سطوح پائین‌تر شروع شود. فرض را بر آن خواهیم داشت که هر ایستگاه تخمین خوبی از تعداد کلّ ایستگاه‌های آمادهٔ ارسال (که آنرا q می‌نامیم) در اختیار دارد و این تخمین را به روشی مثل نظارت بر ترافیک جاری شبکه بدست آورده است.

در ادامه اجازه بدهید سطوح درخت را شماره‌گذاری کرده و رأس آن یعنی گره شمارهٔ ۱ را سطح صفر بنامیم. بدین ترتیب، گره ۲ و ۳ در سطح یک قرار می‌گیرند و شماره گذاری سطوح بهمین نحو ادامه می‌یابد. دقت کنید که هرگره در سطح i، کسر $\frac{1}{2^i}$ از کلّ ایستگاه‌های شبکه را در بر می‌گیرد. اگر q ایستگاه آمادهٔ ارسال، بطور یکنواخت در گره‌ها توزیع شده باشند، میانگین تعداد ایستگاه‌های آمادهٔ ارسال که‌تحت پوشش گرهی خاص در سطح i قرار دارند، معادل $q.2^{-i}$ خواهد بود. بطور حسّی می‌توان انتظار داشت که سطح بهینه‌ای که جستجو باید از آن سطح آغاز شود همان سطحی است، تعداد متوسط ایستگاه‌های آماده در زیر هر گره ۱ باشد، یعنی همان سطحی که در آن $q.2^{-i} = 1$ است. با حلّ این معادله بدست می‌آوریم: $i = \log_2 q$.

دو پژوهشگر بنامهای Bertsekas و Gallager (1992)، نسخه‌های متعدد و پیشرفته‌تری از الگوریتم فوق را ابداع و جزئیات آنها را تشریح کرده‌اند. به عنوان مثال، حالتی را در نظر بگیرید که در آن ایستگاه‌های G و H [تحت پوشش گره ۷] تنها ایستگاه‌های آمادهٔ ارسال هستند. در گره ۱ تصادم رخ خواهد داد، در حالیکه وقتی در برش شماره ۲ نوبت به رقابت اعضای گره ۲ می‌رسد، آن برش خالی خواهد ماند. مجدداً در حین آزمایش گره ۳ تصادم پیش می‌آید؛ (تا اینجا می‌دانیم که دو یا چند ایستگاه در گره ۱ تمایل به ارسال دارند ولی قطعاً در گره ۲ واقع

---

۱. در مبحث ساختمان داده و الگوریتمها، پیمایش عمقی روشی برای پیمایش درخت محسوب می‌شود. ـم

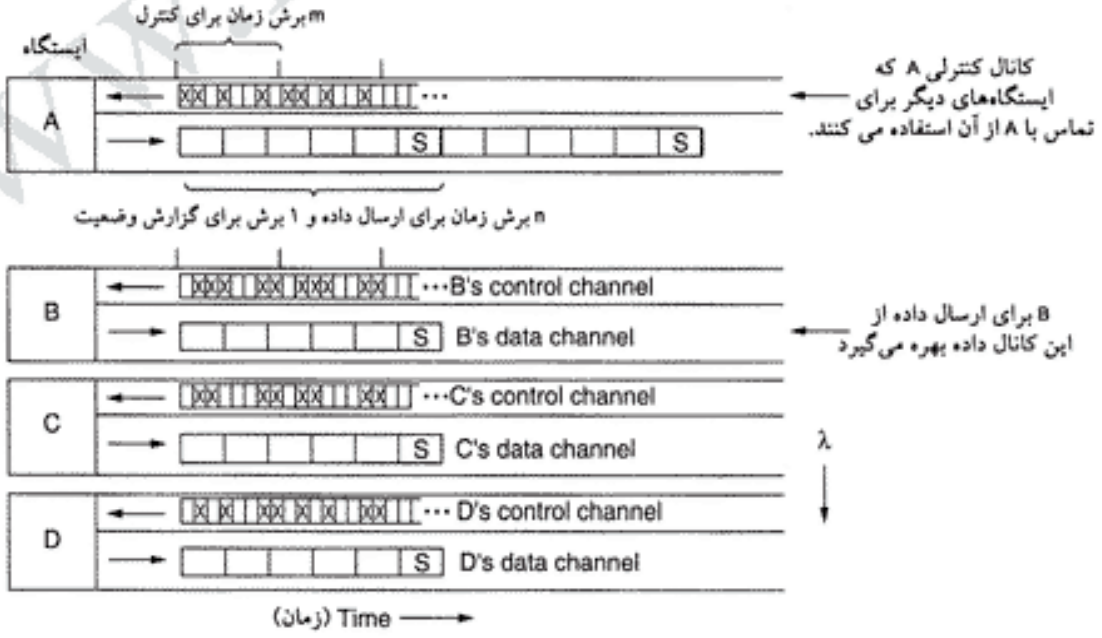
نشده‌اند). آزمایش گره ۳ ناکام مانده و به آزمون گره ۶ می‌انجامد. باز هم در آزمایش گره ۷ تصادم رخ می‌دهد و عاقبت در آخرین تلاش G موفق به ارسال خواهد شد. دو پژوهشگر فوق برای حلّ مشکلات این چنینی، راهکارهایی را معرفی کرده‌اند.

٤-٢-٥ پروتکلهای دسترسی چندگانه مبتنی بر تقسیم طول موج[1]

یک راهکار متفاوت برای تخصیص کانال آنست که آنرا بروشی مثل FDM یا TDM (یا تلفیقی از هر دو) به چندین «زیرکانال» تقسیم کرده و آنها را در صورت نیاز به صورت پویا به ایستگاه‌ها تخصیص بدهیم. چنین الگویی، به طور رایج برروی شبکه‌های محلی با فیبرنوری کاربرد دارد تا با استفاده از طول موجهای مختلف (یعنی فرکانسهای متفاوت)، امکان محاوره و ارتباط همزمان ایستگاه‌ها فراهم آید. در این بخش یکی از این پروتکلها را بررسی خواهیم کرد.(Humblet et al., 1992)

ساده‌ترین راه برای ساختن یک شبکهٔ محلّی کاملاً نوری، آنست که شبیه به شکل ۲ـ۱۰، از ترویج کنندهٔ غیرفعّال با توپولوژی ستاره استفاده نمائیم. در حقیقت دو فیبرنوری منشعب شده از هر ایستگاه به یک استوانهٔ شیشه‌ای وارد شده‌اند. [در این استوانه تمام پرتوهای نوری با هم ممزوج شده، در هم می‌آمیزند.-م] یکی از تارهای فیبر نوری بعنوان خروجی و دیگری بعنوان ورودی ایستگاه از این استوانه منشعب می‌شوند. پرتوی نور خارج شده از یک ایستگاه، در درون استوانه منتشر شده و در ورودی تمام ایستگاهها قابل دریافت و تشخیص است. شبکه‌های نوری غیر فعال با توپولوژی ستاره می‌توانند صدها ایستگاه داشته باشند.

برای آنکه ارسال همزمان چندین ایستگاه میّسر باشد، طیف نوری به چندین کانال (باندهای مختلف با طول موج متفاوت) تقسیم می‌گردد. (شکل ۲ـ۳۱ رابینید.) در این پروتکل که WDMA (دسترسی چندگانه مبتنی بر تقسیم طول موج) نامیده شده به هر ایستگاه دو کانال منتسب می‌شود: یک کانال با پهنای باند باریک که از آن به عنوان کانال کنترل، جهت هماهنگی (سیگنالینگ) با ایستگاه استفاده می‌شود و یک کانال با پهنای باند وسیع که برای ارسال فریم، در اختیار ایستگاه قرار می‌گیرد.



شکل ۴ـ۱۰. دسترسی چندگانه مبتنی بر تقسیم طول موج.

۱ – Wavelength Division Multiple Access Protocol

به نحوی که در شکل ۴-۱۰ دیده می‌شود هر کانال به چندین گروه برش زمانی مستقل (اسلات) تقسیم شده است. اجازه بدهید تعداد برشهای زمانی در کانال کنترل را m و تعداد برشهای زمانی کانال داده را n+1 فرض کنیم؛ n تا از برشهای کانال داده، برای ارسال فریم و آخرین آنها برای گزارش وضعیت خود ایستگاه، کاربرد دارد (در حقیقت، این برش برای گزارش برشهای آزاد در دو کانال به کار می‌آید.) در هر دو کانال [داده و کنترل] دنبالهٔ برشهای زمانی بطور دائم و متناوب تکرار می‌شوند و از برش شمارهٔ صفر آغاز می‌گردد. برش شمارهٔ صفر بگونه‌ای نشانه‌گذاری شده که دیگر ایستگاه‌ها بسادگی قادر به تشخیص آن هستند. تمام کانالها با استفاده از یک سیگنال ساعت سراسری سنکرون می‌شوند.

این پروتکل از سه ردهٔ متفاوت ترافیک حمایت می‌کند: (۱) ترافیک اتصال‌گرا با نرخ ثابت برای عملیاتی نظیر ارسال تصاویر ویدیوئی غیرفشرده (۲) ترافیک اتصال‌گرا با نرخ متغیر برای عملیاتی نظیر انتقال فایل (۳) ترافیک دیتاگرام مثل ارسال بسته‌های UDP.

در دو پروتکل اتصال‌گرا، نظریهٔ آن بوده که وقتی A بخواهد با B ارتباط برقرار کند ابتدا یک فریم کنترلی خاص به نام «فریم تقاضای اتصال» (CONNECTION REQUEST) در یک برش خالی از کانال کنترلی متعلق به B قرار می‌دهد. اگر B این تقاضا را پذیرفت مبادله، از طریق کانال داده انجام می‌شود.

هر ایستگاه دارای دو فرستنده و دو گیرنده به ترتیب زیر است:

۱. یک گیرنده با طول موج ثابت برای گوش دادن به کانال کنترلی خودش
۲. یک فرستندهٔ قابل تنظیم برای ارسال بر روی کانالهای کنترلی دیگر ایستگاه‌ها
۳. یک فرستنده با طول موج ثابت برای انتقال فریمهای داده بر روی خروجی خودش
۴. یک گیرنده با طول موج قابل تنظیم برای انتخاب یکی از فرستنده‌های داده و گوش دادن به آن

به عبارت دیگر هر ایستگاه مدام به کانال کنترلی خودش گوش فرا می‌دهد تا تقاضاهای ارتباط را دریافت کند ولیکن برای دریافت داده‌های دیگران، باید خود را با طول موج فرستنده تنظیم کرده و تطبیق بدهد. تنظیم طول موج به کمک ابزاری به نام «**اینترفیرومتر فابری-پرو یا ماخ-زندر**»[1] انجام می‌شود که در حقیقت نوعی فیلتر نوری است که در آن به غیر از یک باند خاص از طول موجها، بقیه حذف می‌شوند.

حال ببینیم که ایستگاه A، چگونه یک کانال ردهٔ ۲ [یعنی ارسال اتصال‌گرا با نرخ متغیر] با B، مثلاً برای انتقال فایل ایجاد می‌کند. ابتدا A، گیرندهٔ دادهٔ خود را برروی طول موج ایستگاه B تنظیم می‌نماید و انتظار می‌کشد تا به یک «**برش گزارش وضعیت**»[2] (Status Slot) برسد. این برش زمانی، مشخص می‌کند که کدامیک از برشها در کانال کنترلی، خالی و کدام تخصیص داده شده‌اند. بعنوان مثال، در شکل ۴-۱۰ می‌بینیم که از میان هشت برش کنترلی متعلق به ایستگاه B، شماره‌های ۰ و ۴ و ۵ آزاد و بقیه اشغال هستند. (علامت ×بمعنای پربودن برش است.) ایستگاه A یکی از برشهای آزاد کانال کنترل (مثلاً ۴) را انتخاب کرده و پیام «**تقاضای اتصال**» (CONNECTION REQUEST) خود را در آن می‌گذارد. از آنجایی که B بطور مدام بر کانال کنترل خود نظارت دارد این تقاضا رامی‌بیند و برش ۴ را به A انتساب می‌دهد. این انتساب در «برش گزارش وضعیت» از کانال دادهٔ متعلق به ایستگاه B، اعلام می‌شود. وقتی A این اعلام را دریافت می‌کند متوجه می‌شود که اتصالی یک طرفه برایش مهیا است. اگر A تقاضای اتصالی دوطرفه داشته باشد، ایستگاه B نیز باید همین روال را انجام بدهد.

ممکن است درست در زمانی که A سعی می‌کند برش شماره ۴ از کانال کنترل B را برای خود تصرف نماید، C

---

۱. Fabry-Perot or Mach-Zehnder Interferometer

۲. «برش گزارش وضعیت» (Status Slot)، آخرین برش در هر گروه از برشهای زمانی است.

نیز همین کار را بکند. در این حالت هیچکدام موفق نخواهند شد و با نظارت بر «برش گزارش وضعیت» متوجه این شکست می‌شوند. [چون پاسخی در «برش گزارش وضعیت» دریافت نمی‌کنند.] در این حالت هر یک به اندازهٔ یک مقدار زمان تصادفی صبر کرده و از نو شروع می‌نمایند.

در این لحظه، هر یک از طرفین روشی بدون اشکال برای ارسال پیامهای کنترلی کوتاه خود به طرف دیگر در اختیار دارند. برای انجام عملیات انتقال فایل، اکنون A می‌تواند پیغام کنترلی کوتاهی برای B مثلاً بدین مضمون بفرستد: «لطفاً داده‌های من را در برش داده شماره ۳ دریافت کنید؛ در این برش یک فریم داده برای شما ارسال شده است!» وقتی B این پیام کنترلی را دریافت می‌کند گیرنده خود را با طول موج کانال خروجی A تنظیم می‌کند. بسته به پروتکل لایه بالاتر، B نیز می‌تواند در صورت تمایل از چنین مکانیزمی برای برگرداندن پیام‌های تصدیق (ACK) استفاده نماید.

دقت کنید که مسئله دیگری نیز ممکن است بوجود بیاید و آن هم اینکه اگر A و C اتصالی با B داشته باشند و هر کدام بطور همزمان به B اعلام کند که به برش شماره ۳ مراجعه کند. B تقاضای یکی از آنها را بطور تصادفی انتخاب کرده و دیگری قادر به ارسال نخواهد شد.

برای ترافیک با نرخ ثابت، گونه دیگری از این پروتکل به کار گرفته می‌شود: وقتی A تقاضای ایجاد یک اتصال می‌کند، بطور لحظه‌ای فریمی را بدین مضمون برای B می‌فرستد: «آیا مجاز به ارسال دائم در برش شماره ۳ برای شما هستم؟» اگر B قادر به پذیرش چنین تقاضایی باشد (یعنی هیچ قرار قبلی برای این برش نگذاشته باشد)، یک اتصال با پهنای باند تضمین‌شده، ایجاد خواهد شد. در غیر این صورت A می تواند پیشنهاد دیگری را براساس خالی بودن برشهای دیگر ارائه بدهد. برای ترافیک رده سوّم (یعنی ترافیک دیتاگرام) نیز از گونهٔ متفاوتی استفاده می‌شود: به جای نوشتن پیغام «تقاضای اتصال» در یک برش کنترلی، پیام DATA FOR YOU IN SLOT 3 را بر روی کانال کنترل قرار می‌دهد (به مضمون آنکه در برش ۳ داده‌ای بر شما وجود دارد). اگر B در خلال برش شمارهٔ ۳ از کانال داده آزاد باشد، انتقال انجام می‌شود و در غیر این صورت فریم از دست خواهد رفت. بدین ترتیب نیاز به برقراری هیچ اتصالی نخواهد بود.

گونه‌های متفاوتی از این پروتکل قابل اجرا است. مثلاً به جای آنکه هر ایستگاه بطور مستقل برای خودش کانال کنترل داشته باشد، یک کانال کنترلی واحد بین همهٔ ایستگاه‌ها مشترک باشد. به هر ایستگاه نیز مجموعه‌ای از برشهای هر گروه منتسب می‌شود تا بتوان چندین کانال مجازی را برروی یک کانال فیزیکی واحد مالتی‌پلکس کرد. همچنین این امکان وجود دارد که فقط از یک فرستندهٔ قابل تنظیم و یک گیرندهٔ قابل تنظیم در هر ایستگاه استفاده شود و کانال واحد هر ایستگاه به m برش کنترلی و بدنبال آن n+1 ، برش داده تقسیم گردد. اشکال این روش آنست که ایستگاه‌های فرستنده مجبورند برای در اختیار گرفتن کانال، زمان بیشتری را منتظر بمانند و از طرفی دنبالهٔ فریمهای داده با فاصلهٔ بیشتری از هم ارسال می‌شوند چراکه فریم کنترلی مابین آنها قرار گرفته است.

تاکنون پروتکلهای WDMA بی‌شماری پیشنهاد و پیاده‌سازی شده که در بسیاری از جزئیات با هم متفاوت هستند. برخی از آنها دارای یک کانال کنترل واحد هستند در حالیکه برخی دیگر چندین کانال کنترلی دارند. بعضی از آنها تاخیر انتشار را به حساب آورده‌اند در حالیکه بعضی از آن چشمپوشی کرده‌اند. برخی از آنها زمان تنظیم [فیلتر گیرنده یا فرستنده] را به عنوان بخشی مشخص از مدل خود در نظر گرفته‌اند در حالیکه برخی از آن صرفنظر کرده‌اند. این پروتکلها از دیدگاه پیچیدگی پردازش ، ظرفیت و بهرهٔ مفید و قابلیت گسترش نیز با یکدیگر متفاوتند. به سیستمی که از تعداد زیادی فرکانس استفاده می‌کند، اصطلاحاً DWDM (Dense Wavelength Division Multiplexing) اطلاق می‌شود. برای آگاهی بیشتر به مراجع ذیل مراجعه کنید:

(Bogineni et al., 1993; Chen, 1994; Goralski, 2001; Kartalopoulos, 1999;Levine & Akyidiz, 1995)

٤-۲-۶ پروتکلهای بی‌سیم برای شبکهٔ محلی

به موازات رشد تعداد دستگاه‌های محاسباتی و مخابراتی قابل حمل‌ونقل (همراه)، تقاضا برای اتصال آنها به دنیای خارج نیز افزایش یافته است. اگرچه حتی نخستین سری تلفنهای همراه امکان اتصال به تلفن دیگر را داشتند ولیکن کامپیوترهای کیفی، فاقد چنین قابلیتی بودند. چیزی نگذشت که مودمها به عنوان یک ابزار معمولی برروی کامپیوترهای همراه جا خوش کردند ولیکن برای برقراری ارتباط، می‌بایست سیم این کامپیوترها را به پریزهای دیواری تلفن وصل کرد. نیاز به سیم جهت اتصال به یک شبکهٔ ثابت ، بدین معناست که کامپیوترها قابل حمل‌ونقل هستند ولی «همراه» (Mobile) تلقی نمی‌شوند.

برای تحقق معنای واقعی «همراه بودن» ، کامپیوترهای کیفی نیاز به ارتباط از طریق سیگنالهای رادیوئی (یا مادون قرمز) داشتند. در چنین شرایطی، کاربران می‌توانند در حین گردش یا قایقرانی، نامه‌های خود را بخوانند یا نامه بفرستند. به گونه‌ای که در بخش ۱-۵-۴ اشاره شد، کامپیوترهای کیفی که از طریق سیگنالهای رادیوئی با یکدیگر مخابرهٔ اطلاعات می‌کنند ، شبکهٔ محّلی بی‌سیم تلقی می‌شوند. اینگونه شبکه‌های محلی تفاوتهای عمده‌ای با شبکه‌های محلی رایج دارند و به پروتکلهای خاصی در زیرلایهٔ MAC نیازمندند. در این بخش برخی از این پروتکلها را بررسی خواهیم کرد. برای آگاهی بیشتر در مورد شبکه‌های محلی بی‌سیم مراجع (Geier, 2002; O'Hara and Petrick, 1999) مفیدند.

پیکربندی رایج برای شبکه‌های محّلی بی‌سیم بدین نحو است که در یک ساختمان اداری تعدادی **ایستگاه ثابت** (که نقاط دسترسی -Access Point- نامیده می‌شوند) نصب می‌شود. تمام این ایستگاه‌های ثابت از طریق فیبرنوری یا سیم مسی به یکدیگر متصل هستند. اگر توان انتقال رادیویی ایستگاه‌های ثابت و کامپیوترهای کیفی بگونه‌ای تنظیم شود که محدوده‌ای حدود ۳ تا ۴ متر را پوشش بدهد هر اتاق در ساختمان نقش یک «سلول» را ایفاء خواهد کرد و کلّ ساختمان همانند سیستمهای معمولی تلفن سلولی (یعنی شبکهٔ تلفن همراه که در فصل ۲ بدان پرداختیم) عمل می‌کند ولی برخلاف سیستم تلفن سلولی ، هر سلول تنها دارای یک کانال واحد است که کل پهنای باند موجود را در برگرفته و تمام ایستگاه‌های درون آن سلول را پوشش خواهد داد. عموماً پهنای باند این شبکه بین ۱۱ تا ۴۵ مگابیت بر ثانیه است.

در توضیحات زیر برای ساده‌تر شدن بحث، فرض را بر آن گذاشته‌ایم که تمام فرستنده‌های رادیوئی، دارای برد ثابت و محدودی هستند. وقتی گیرنده‌ای در برد دو ایستگاه فعّال و در حال ارسال قرار بگیرد سیگنال دریافتی او ، مصدوم و بلااستفاده خواهد بود؛ درک این حقیقت که در اغلب شبکه‌های محلی بی‌سیم، تمام ایستگاه‌ها الزاماً در برد یکدیگر قرار ندارند، بسیار مهم است و منجر به پیچیدگی‌هائی خواهد شد. به علاوه هرگاه شبکهٔ محلی بی‌سیم در داخل ساختمان قرار گرفته باشد وجود دیوار بین ایستگاه‌ها می‌تواند تاثیر مخربی برروی برد هر ایستگاه داشته باشد.

یک روش ناشیانه برای تخصیص کانال در شبکهٔ محلی بی‌سیم، بکارگیری روش CSMA است: ایستگاه‌ها به کانال گوش بدهند و در صورتی که هیچ ایستگاه دیگری در حال ارسال نبود، انتقال فریم انجام شود. مشکلی که در این پروتکل وجود دارد [در حالیکه در شبکه‌های مبتنی بر کابل وجود ندارد] آنست که در محیط بی‌سیم ، تداخل امواج و تصادم فقط در گیرنده اهمیت دارد نه در فرستنده! برای آنکه به ماهیت این مشکل پی‌ببریم به شکل ۴-۱۱ که در آن چهار ایستگاه بی‌سیم ترسیم شده نگاهی بیندازید. اینکه کدام ایستگاه ثابت و کدامیک کامپیوتر کیفی [همراه] هستند، برای هدفی که در پیش داریم اهمیتی ندارد. برد رادیوئی ایستگاه‌ها به گونه‌ای است که A و B در

شکل ۴-۱۱. یک شبکهٔ محلی بی‌سیم (الف) A در حال ارسال (ب) B در حال ارسال.

برد یکدیگر هستند و سیگنال آنها می‌توانند با یکدیگر تداخل کرده، خراب شود. C نیز می‌تواند با B و D تداخل سیگنال داشته باشد ولی با A تداخل ندارد [چون در برد او نیست].

ابتدا فرض کنید که A در حال ارسال برای B ، است: (شکل ۴-۱۱-الف) اگر C کانال را شنود کند هیچ چیزی از A نمی‌شنود چون در برد A نیست و بدین ترتیب به اشتباه نتیجه می‌گیرد که می‌تواند برای B ارسال داشته باشد. اگر C ارسال خود را آغاز نماید در ایستگاه B [با سیگنال ارسالی از A] تداخل کرده و فریم رسیده از A را نابود خواهد کرد. این مشکل که یک ایستگاه قادر نیست حضور یک رقیب را برروی کانال تشخیص بدهد، «**مشکل ایستگاه پنهان**» (Hidden Station Problem) نامیده می‌شود و از آنجا ناشی می‌شود که ایستگاه‌ها از هم دور بوده و سیگنال‌های یکدیگر را نمی‌شنوند.

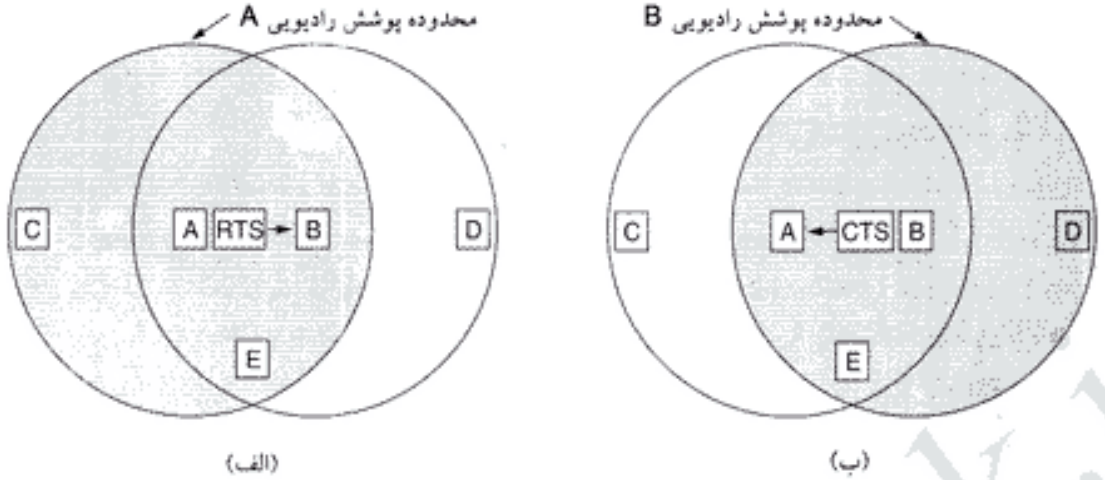
اکنون حالت برعکس شرایط فوق را بررسی می‌کنیم: به نحوی که در شکل ۴-۱۱-ب می‌بینید، B در حال ارسال برای A است. اگر C به شنود کانال بپردازد متوجه می‌شود که کانال اشغال است و به غلط نتیجه می‌گیرد که نباید برای D ارسال داشته باشد در حالیکه ارسال برای D هیچ اشکالی ندارد چرا که A و D در ناحیه‌ای دور از یکدیگر قرار گرفته‌اند و تصادمی پدید نخواهد آمد. این مشکل به نام «**مشکل ایستگاه آشکار**» (Exposed Station Problem) نامیده می‌شود.

مسئله اصلی آنست که یک دستگاه قبل از شروع به ارسال می‌خواهد بداند که آیا در پیرامون گیرنده، فعالیتی وجود دارد یا خیر؟ روش CSMA صرفاً می‌تواند در مورد فعالیت پیرامون فرستنده تشخیص خود را ارائه بدهد. به خاطر داشته باشید که بر روی سیم، سیگنالها به تمامی ایستگاه‌ها منتشر می‌شوند و طبعاً در آن واحد، فقط یک ایستگاه می‌تواند ارسال داشته باشد. در سیستمی که مبتنی بر امواج رادیوئی با برد کوتاه است چندین ایستگاه می‌توانند بطور همزمان ارسال داشته باشند البته مشروط به آنکه ایستگاه‌های مقصد ، متفاوت و در محدودهٔ برد یکدیگر نباشند.

روش دیگری جهت اندیشیدن به این مشکل، تجسم اداره‌ای است که کارمندان آن کامپیوترهای کیفی بهمراه کارت شبکهٔ بی‌سیم در اختیار دارند. فرض کنید کارمندی مثل لیندا می‌خواهد پیامی را برای میلتون بفرستد. کامپیوتر لیندا محیط پیرامون خود را شنود کرده و چون فعالیتی را تشخیص نمی‌دهد ارسال را شروع می‌کند. غافل از آنکه در محل دفتر کار میلتون، تصادم رخ خواهد داد زیرا شخص ثالثی هم‌اکنون در حال ارسال برای اوست و از آنجا که دفتر کار آن شخص از لیندا دور بوده، فعالیت او تشخیص داده نشده است.

MACAW, MACA

یکی از اولین پروتکلهای طراحی شده برای شبکه‌های محلی بی‌سیم ، پروتکل MACA (پروتکل دسترسی چندگانه با اجتناب از تصادم) است. (Karn,1990) ایدهٔ اصلی در این روش آنست که فرستنده به نحوی گیرنده را تحریک به ارسال یک فریم کوتاه برای ایستگاه‌های پیرامون خود کند تا آنهایی که در برد او هستند و این فریم کوتاه را می‌شنوند ، از ارسال اطلاعات در خلال زمان دریافت فریم ، خودداری کنند. روش MACA در شکل ۴-۱۲ نشان داده شده است.



شکل ۴-۱۲. پروتکل MACA. (الف) A در حال ارسال یک فریم RTS به B.(الف) B در حال ارسال فریم پاسخ CTS به A.

حال بررسی کنیم که چگونه A فریمی را برای B می‌فرستد. ایستگاه A کارش را با ارسال یک فریم کوتاه به نام RTS[1] برای B شروع می‌کند. (شکل ۴-۱۲-الف) درون این فریم کوتاه ۳۰ بایتی، طول کلّ فریم داده‌ای که قرار است در آینده ارسال گردد مشخص شده است. B در پاسخ ، فریمی به نام CTS[2] ارسال می‌نماید. فریم CTS نیز طول فریم داده‌ای را که قرار است ارسال شود، مشخص می‌کند؛ (شکل ۴-۱۲-ب). پس از دریافت فریم CTS ، ایستگاه A می‌تواند ارسال خود را آغاز نماید.

حال ببینیم ایستگاه‌هائی که این فریمها را می‌شنوند چه عکس‌العملی نشان می‌دهند. هر ایستگاهی که RTS را می‌شنود به A نزدیک است و باید به اندازه‌ای صبر کند تا پیام CTS ، بدون تداخل و تصادم باز گردد. هر ایستگاه که CTS را می‌شنود، نزدیک به B (گیرندهٔ فریم داده) است و باید در خلال ارسال فریم کامل توسط A ، ساکت بماند. از آنجاکه طول فریم داده‌ای که در آینده قرار است ارسال شود، در فریم CTS مشخص شده، ایستگاه‌های شنوندهٔ CTS نیز می‌توانند زمان سکوت و انتظار، را تخمین بزنند.

در شکل ۴-۱۲ ایستگاه C در برد A است ولی در برد B نیست لذا RTS منتشره از A را می‌شنود ولی CTS ارسالی از B را نمی‌شنود. این ایستگاه مجاز است پس از آنکه به اندازهٔ زمان ارسال فریم CTS منتظر ماند، همزمان با ارسال فریم دادهٔ A ، او هم به ارسال خود مشغول باشد. در سمت مقابل، D در برد B هست ولی در برد A قرار ندارد، فلذا RTS را نمی‌شنود در حالیکه CTS را می‌شنود. شنیدن CTS بدین معناست که ایستگاه در نزدیکی ایستگاه گیرنده واقع شده و ایستگاه‌های شنوندهٔ CTS باید ارسال خود را آنقدر به تعویق بیندازند تا ارسال فریم مورد انتظار به پایان برسد. ایستگاه E هر دو فریم کنترلی RTS و CTS را می‌شنود و همانند D بایستی منتظر بماند تا ارسال فریم خاتمه یابد.

با تمام این تمهیدات، باز هم وقوع تصادم محتمل است. به عنوان مثال اگر B و C هر دو بطور همزمان فریم RTS برای A بفرستند، تصادم رخ می‌دهد و فریم از بین خواهد رفت. در صورت بروز تصادم، فرستندهٔ ناموفق (یعنی فرستنده‌ای که به مدّت زمان مشخصی پس از ارسال RTS ، فریم CTS دریافت نکند) به اندازهٔ یک زمان تصادفی صبر کرده و از نو تلاش می‌کند. الگوریتم انتظار و تکرار مجدّد به نام «الگوریتم عقبگرد نمایی» شهرت دارد و در مطالعهٔ شبکهٔ اترنت آنرا تشریح خواهیم کرد.

به منظور افزایش کارآیی پروتکل MACA ، پژوهشگری به نام Bharghavan و گروه همکار او (۱۹۹۴) با

---

۱. تقاضا برای ارسال یا Request to Send
۲. مجوز ارسال یا Clear To Send

بررسی نتایج مطالعات شبیه‌سازی، اصلاحاتی را پیشنهاد کرده و پروتکل جدید را MACAW[1] نامیدند. در بدو کار آنها متوجه شدند که اگر در لایهٔ پیوند داده ، پس از دریافت یک فریم سالم، پیام اعلام وصول (ACK) آن ارسال نگردد، طبعاً فریمهای از بین‌رفته، ارسال مجدّد نخواهد شد تا آنکه مدتی بعد و در لایهٔ انتقال، عدم وجود آنها کشف شود؛ این زمان انتظار بسیار طولانی و وقتگیر است و کارآیی پروتکل را بشدت کاهش خواهد داد. آنها این مشکل را با معرفی یک فریم جدید ACK که پس از دریافت موفق یک فریم داده ارسال می‌شود، حلّ کردند. همچنین آنها متوجه شدند که برخی از ویژگیهای CSMA باز هم در این شبکه قابل استفاده است؛ مثلاً برای آنکه یک ایستگاه ، همزمان با ایستگاه دیگری برای یک مقصد مشابه ، فریم RTS نفرستد ، بهتر است قبل از ارسال RTS کانال شنود شود؛ بدین ترتیب قابلیت شنود سیگنال حامل نیز به پروتکل اضافه شد. به علاوه آنها تصمیم گرفتند که الگوریتم «عقبگرد نمایی» را به جای آنکه برای هر ایستگاه به کار ببرند بطور مستقل برای یک استریم داده (که بین یک زوج مبداء و مقصد جریان دارد) اعمال نمایند. در آخر آنها به این پروتکل مکانیزمهائی افزودند تا ایستگاه‌ها با یکدیگر اطلاعاتی در خصوص ازدحام (Congestion) ردّ و بدل کنند. همچنین الگوریتم «عقب‌گرد نمایی» را بگونه‌ای تغییر دادند تا در خصوص مشکلات موقت، عکس‌العمل شدید از خودش نشان ندهد و بدین ترتیب کارآئی سیستم بهبود یابد.

۳.٤ اترنت

تا اینجا پروتکلهای تخصیص کانال را بصورت کلّی و اجمالی مطالعه کردیم؛ اکنون زمان آن فرا رسیده تا این قواعد را برروی سیستمهای واقعی و خصوصاً شبکه‌های LAN ، اعمال نمائیم. همانگونه که در بخش ۳.۵.۱ اشاره شد، IEEE تعدادی از شبکه‌های محلی و بین‌شهری را با نام IEEE 802 استانداردسازی کرده است. در فهرست شکل ۳۸.۱ دیدیم که برخی از این شبکه‌ها هنوز وجود دارند و برخی دیگر به تاریخ پیوسته‌اند. شاید آنهایی که به «نظریهٔ تناسخ روح» معتقدند بتوانند اینگونه بیندیشند که آقای چارلز داروین برگشته و به عنوان یکی از اعضای کمیتهٔ IEEE در حال حذف، پالایش و تکمیل زنجیرهٔ استانداردهاست!!! از بین استانداردهای باقیمانده از گذشته، می‌توان به 802.3 (اترنت) و 802.11 (شبکهٔ محلی بی‌سیم) اشاره کرد. هنوز زود است که در خصوص 802.15 (Blue tooth) و 802.16 (شبکهٔ بین شهری بی‌سیم) قضاوت کنیم؛ در این مورد می‌توانید به ویرایش پنجم این کتاب مراجعه کنید! لایهٔ فیزیکی و زیرلایهٔ MAC در شبکه‌های 802.3 و 802.11 تفاوت بنیانی دارند ولی این تفاوتها در زیرلایهٔ منطقی (تعریف شده در 802.2 LLC) ، همگرا خواهد شد و بدین ترتیب هر دوی آنها، واسط (Interface) مشابهی با لایهٔ شبکه دارند. [LLC فوقانی‌ترین زیرلایه از لایهٔ پیوند داده‌هاست و وظیفه دارد تفاوتها و ناهمگونیهای اجتناب‌ناپذیر زیرلایه‌های پایین را از دید لایهٔ شبکه مخفی نگاه دارد تا سرویسهایی که به این لایه ارائه می‌شود استاندارد و یکسان باشد. -م]

در خصوص اترنت در بخش ۱.۵.۳ توضیحاتی داده شده است؛ لذا آن مفاد را در اینجا تکرار نخواهیم کرد. در عوض، بر روی جزئیات تخصصی اترنت، پروتکلهای مرتبط با آن و پیشرفتهای اخیر در زمینهٔ اترنت سرعت‌بالا (گیگابیت اترنت) متمرکز خواهیم شد. از آنجایی که «اترنت» و IEEE 802.3 به غیر از دو تفاوت جزئی که به آنها اشاره خواهیم کرد، از بقیهٔ جهات یکسان هستند، بسیاری از افراد این دو شبکه را معادل هم در نظر گرفته و ما نیز این دو را یکی فرض خواهیم کرد. برای اطلاعات بیشتر در خصوص اترنت از مراجع زیر استفاده کنید:

Breyer and Riley, 1999; Seifert, 1998; and Spurgeon 2000)

---

۱. Multiple Access with Collision Avoidance for Wireless networks

٤-٣-١ کابل‌کشی اترنت

از آنجایی که نام اترنت در اصل برگرفته از واژهٔ «اتر» است که به کابل اشاره دارد ما نیز توضیحات خود را از کابل شروع خواهیم کرد. در این شبکه چهار نوع کابل که فهرست آنها در شکل ۴-۱۳ آمده، رایج هستند:

مزایا	تعداد گره در هر قطعه	حداکثر طول قطعه	نوع کابل	نام کابل
کابل اصلی و اولیه (از رده خارج)	100	500 m	Thick coax	10Base5
به هاب نیازی نیست	30	185 m	Thin coax	10Base2
ارزان‌ترین سیستم	1024	100 m	Twisted pair	10Base-T
بهترین انتخاب برای مابین ساختمانها	1024	2000 m	Fiber optics	10Base-F

شکل ۴-۱۳. رایجترین انواع کابل‌کشی اترنت.

از منظر تاریخ، نخستین نوع کابل در اترنت ، 10Base5 بود که عموماً به «اترنت ضخیم» شهرت داشت. این کابل به شیلنگهای زردرنگ باغبانی شبیه است و هر ۲/۵ متر برروی آن علامتی گذاشته شده تا محل انشعاب تزریقی مشخص باشد. (البته طبق استاندارد 802.3 ، زردبودن کابل الزامی نیست ولی پیشنهاد شده است.) اتصال به کابل عموماً از طریق «**انشعاب تزریقی**» (Vampire Tap) انجام می‌شود که در آن یک سوزن به دقت در مرکز کابل کوآکسیال فرو می‌رود. نماد 10Base5 بدین معنی است که شبکه با نرخ 10Mbps کار می‌کند، از سیگنالینگ باند پایه (Baseband) بهره گرفته و طول حداکثر یک قطعه کابل ۵۰۰ متر است. گونهٔ دیگری کابل با نام 10Broad36 ، در باند وسیع (Broadband) طراحی شد ولی هیچگاه به بازار نیامد و عملاً از صحنه محو گردید. در صورتی که کانال از نوع کابل کوآکس باشد، عددی که بعد از کلمهٔ Base ظاهر می‌شود، حداکثر طول کابل را بر مبنای ۱۰۰ متر مشخص می‌کند.

نوع دوم کابل‌کشی اترنت 10Base2 یا «اترنت نازک» نام داشت که برخلاف کابل قبلی (که شبیه به شیلنگ باغبانی و غیرقابل‌انعطاف بود) به راحتی خم می‌شد. برای اتصال به این نوع کابل، به جای استفاده از انشعاب تزریقی، می‌توان از کانکتورهای BNC معمولی و ایجاد یک اتصال بشکل T ، بهره گرفت. کانکتورهای BNC بسیار قابل‌اعتماد و کاربرد آنها ساده‌تر است. کابلهای اترنت نازک نیز، بسیار ارزان و نصب آن راحت است ولیکن حداکثر طول کانال به ۱۸۵ متر کاهش یافته و به هر قطعه کابل حداکثر می‌توان ۳۰ ایستگاه متصل کرد.

تشخیص طول بیش از اندازهٔ کابل ، انشعابات بد ، شل‌شدن اتصالات یا هرگونه پارگی در جائی از آن، از اساسی‌ترین مشکلات این دو نوع کابل محسوب می‌شوند [زیرا بروز یکی از این اشکالات کلّ شبکه از کار خواهد انداخت.] به همین دلیل تکنیکی برای تشخیص این معایب ابداع شده است: یک پالس الکتریکی با شکل معمولی به درون کابل تزریق می‌شود. اگر این پالس به یک مانع [پارگی] یا به انتهای کابل برخورد کند، یک سیگنال بازتاب (Echo) تولیده شده و باز خواهد گشت. با اندازه‌گیری دقیق زمان بین ارسال پالس و زمان دریافت بازتاب آن، پیدا کردن محل تقریبی عامل بازتاب [محل خرابی] کشف خواهد شد. به این روش اصطلاحاً «**بازتاب‌سنجی در حوزهٔ زمان**» (Time Domain Reflectometry) گفته می‌شود.

مشکلات تشخیص محل پارگی در کابل ، باعث شد که الگوی متفاوتی در سیم‌کشی این نوع شبکه به کارگرفته شود؛ در روش جدید هر ایستگاه یک کابل اختصاصی دارد که آنرا به یک هاب مرکزی متصل می‌کند. این هاب اتصال الکتریکی تمام ایستگاه‌ها را از درون ، برقرار می‌سازد. معمولاً این سیم‌ها، از نوع زوج سیم‌های معمولی خطوط تلفن هستند زیرا در ساختمان‌های اداری این سیم‌کشی از قبل وجود دارد و تعداد زیادی از این زوج سیمها بلااستفاده رها شده‌اند. به این روش سیم‌کشی اصطلاحاً 10BaseT گفته می‌شود. هابها ترافیک داده‌های ورودی را

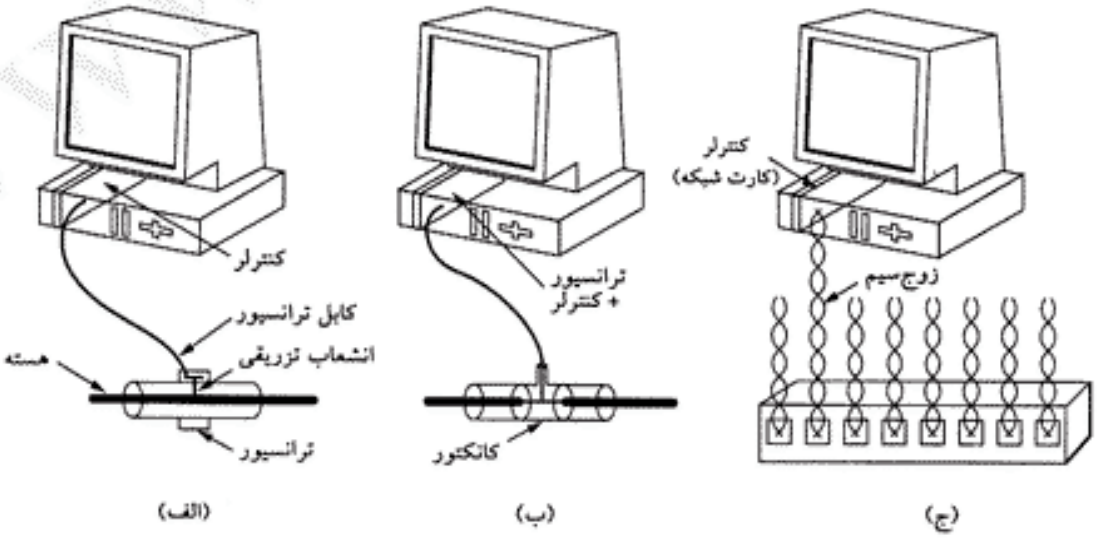
بافر نمی‌کنند [بلکه فقط ایستگاهها را به کانال مشترک متصل می‌نمایند]. در ادامهٔ همین فصل نسخهٔ پیشرفته‌تر این ایده (یعنی سوئیچها) را تشریح خواهیم کرد که قادرند ترافیک ورودی را بافر کنند.

این سه روش سیم‌کشی در شکل ۴-۱۴ به تصویر کشیده شده است. در سیم‌کشی 10Base5 ، مدار **ترانسیور**[1] ، باید بدقت و با اطمینان به دور کابل اصلی، محکم شود تا اتصال آن با هستهٔ مرکزی کابل برقرار گردد. مدار ترانسیور دارای مدار الکترونیکی کوچکی است که حضور سیگنال حامل و بروز تصادم را تشخیص می‌دهد و در صورتی که متوجه بروز تصادم شود یک سیگنال خاص و غیرمعتبر بر روی کابل می‌گذارد تا مطمئن شود بقیه نیز از تصادم مطلع شده‌اند.

در 10Base5 ، یک کابل بنام ترانسیور (کابل اتصال) ارتباط کارت واسط ایستگاه و کابل اصلی را برقرار می‌کند. طول این قطعه کابل، می‌تواند تا ۵۰ متر باشد و درون آن ۵ جفت سیم زره‌دار (Shielded) وجود دارد. دو جفت از آنها برای داده‌های ورودی و خروجی از ایستگاه است. دو جفت نیز برای سیگنالهای کنترلی ورودی/خروجی به کارگرفته می‌شوند. زوج پنجم که ممکن است از آن استفاده نشود برای تغذیهٔ مدار الکترونیکی ترانسیور کاربرد دارد. همچنین، بعضی از ترانسیورها اجازه می‌دهند تا حداکثر هشت کامپیوتر نزدیک به هم، بدانها متصل شود تا تعداد ترانسیورهای مورد نیاز کاهش یابد.

کابل ترانسیور از یک طرف به کارت شبکه در درون کامپیوتر متصل است؛ کارت شبکه شامل یک تراشهٔ کنترلر است که ارسال یا دریافت فریم به ترانسیور را بر عهده دارد. این کنترلر وظیفه دارد داده‌ها را در قالب یک فریم مناسب سازماندهی کند. همچنین محاسبهٔ کدهای کشف خطا برای فریمهای خروجی و ارزیابی صحّت فریمهای ورودی بر عهدهٔ همین کنترلر است. برخی از این تراشه‌ها دارای مقداری بافر هستند تا فریم‌های ورودی را جهت ارسال به صفّ نمایند؛ همچنین قادرند داده‌ها را بروش DMA به حافظه اصلی کامپیوتر منتقل کنند و در ضمن برخی از عملیات مدیریت شبکه را نیز انجام می‌دهند.

در سیم‌کشی 10Base2 ، اتصال کامپیوتر با کابل بکمک یک کانکتور BNC معمولی (از نوع T-شکل) انجام می‌شود. در این نوع سیم‌کشی، بخش الکترونیکی ترانسیور برروی بُرد کنترلر قرار گرفته است و بدین ترتیب هر ایستگاه ترانسیور خود را دارد.



شکل ۴-۱۴. سه روش کابل‌کشی اترنت (الف) 10Base5 (ب) 10Base2 (ج) 10Base-T.

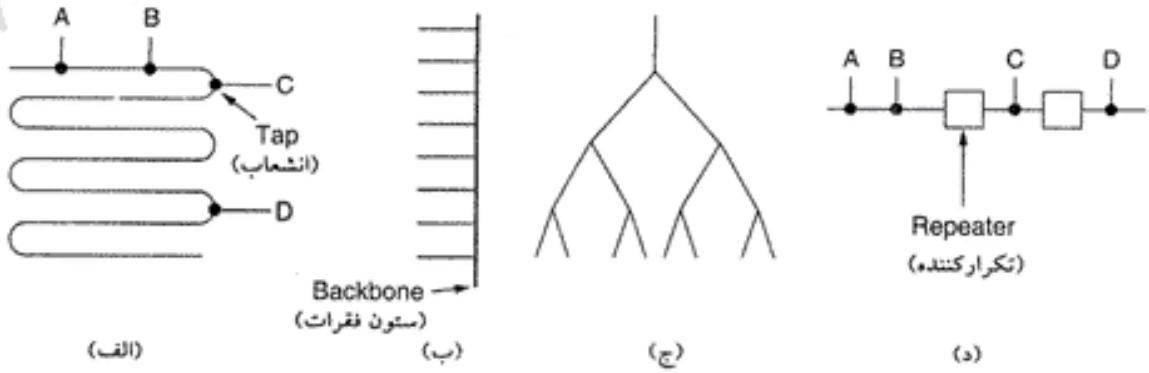
۱. ترانسیور (Tranciever) قطعه‌ای الکترونیکی است که اتصال کابل نازک متصل به ایستگاه و کابل ضخیم اصلی را برقرار می‌کند. ـم

در ساختار 10Base-T ، هیچ کابلی مشترکی وجود ندارد بلکه فقط یک **هاب** (جعبه‌ای پر از مدارات الکترونیک) وجود دارد که تمام ایستگاه‌ها با یک کابل اختصاصی (غیرمشترک) بدان متصل می‌شوند. حذف یا اضافهٔ یک ایستگاه بدین ساختار بسیار ساده و هرگونه پارگی در کانال به راحتی قابل کشف است. اشکال ساختار 10Base-T آنست که طول کابل متصل به هاب ، حداکثر می‌تواند صد متر و در صورت استفاده از کابل باکیفیت و گرانی مثل Cat 5 ، حداکثر دویست متر باشد. علیرغم این محدودیت، ساختار 10Base-T بدلیل سادگی نصب و پشتیبانی و همچنین استفاده از سیم‌کشی موجود ساختمانها، بسیار رایج شده است. نسخه سریعتر 10Base-T یعنی 100Base-T در ادامهٔ همین فصل تشریح خواهد شد.

روش چهارم کابل‌کشی ، 10Base-F است که در آن از فیبرنوری بهره گرفته شده است. این گزینه بدلیل هزینهٔ بالای اتصالات و پایانه‌های[1] مورد نیاز، بسیار گران است ولی در عوض ایمنی بسیار بالائی در مقابل نویز دارد و انتخاب مناسبی برای کابل‌کشی بین ساختمانها یا هابهای دور از هم، به شمار می‌رود. در این روش ، رشته‌هائی با طول کیلومتر نیز مجاز هستند. همچنین در این روش ضریب امنیت اطلاعات بسیار بالاست چراکه انشعات گرفتن از این کانال [و استراق سمع داده‌ها] بسیار دشوارتر سیم‌های مسی است.

شکل ۴-۱۵ روشهای مختلف کابل‌کشی ساختمان را نشان می‌دهد. در شکل ۴-۱۵-الف یک کابل واحد، اطاق به اطاق کشیده شده و هر ایستگاه در نزدیکترین نقطه بدان متصل شده است. در شکل ۴-۱۵-ب یک رشتهٔ عمودی در نقش ستون فقرات از طبقهٔ همکف تا بام کشیده شده و در هر طبقه کابلهای افقی از طریق تقویت کننده‌های خاص (بنام تکرارکننده) به این ستون فقرات متصل شده‌اند. در برخی از ساختمانها کابلهای افقی، نازک و کابل ستون‌فقرات، از نوع ضخیم انتخاب می‌شود. رایجترین ساختار، توپولوژی درختی است که در شکل ۴-۱۵-ج دیده می‌شود.

برای هر یک از روشهای کابل‌کشی اترنت، طول هر قطعه کابل نباید از یک مقدار حداکثر تجاوز کند. برای شبکه‌های وسیع می‌توان شبیه به شکل ۴-۱۵-د از چندین قطعه کابل که توسط «تکرارکننده» بهم متصل شده‌اند، استفاده کرد. تکرارکننده یک ابزار در لایهٔ فیزیکی است داده‌ها را دریافت، تقویت (بازتولید) و مجدداً ارسال می‌کند. از دیدگاه نرم‌افزار، مجموعه‌ای از قطعات کابل که از طریق تکرارکننده به هم متصل شده‌اند هیچ تفاوتی با یک شبکه با کابل یکپارچه ندارد (مگر اندکی تاخیر اضافی که توسط تکرارکننده‌ها تحمیل می‌شود.) یک سیستم ممکن است از چندین قطعه کابل و چند تکرارکننده تشکیل شده باشد ولیکن فاصله بین هر دو ترانسیور نبایستی از ۲/۵ کیلومتر تجاوز کند و در مسیر بین هر دو ترانسیور نباید بیش از ۴ تکرارکننده موجود داشته باشد.



شکل ۴-۱۵ توپولوژیهای مختلف کابل (الف) خطی (ب) ستون فقرات (ج) درختی (د) چندبخشی.

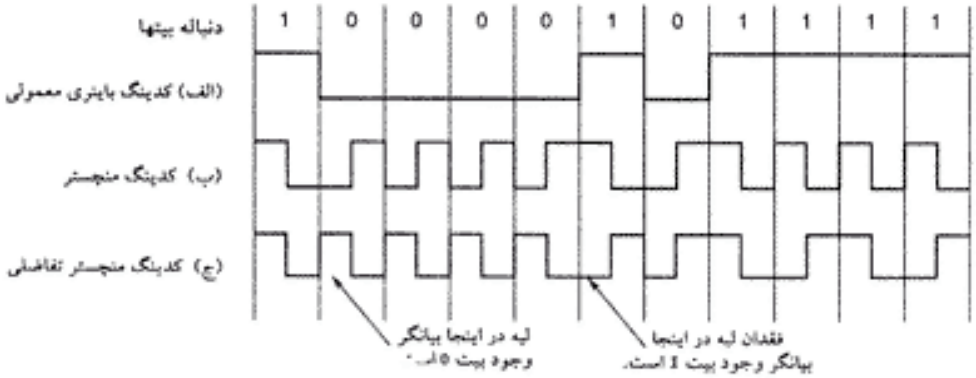
۱. Terminators

٤-٣-٢ کدینگ منچستر

در هیچیک از نسخه‌های مختلف اترنت، از روش معمولی کدینگ (یعنی صفر ولت برای بیت 0 و ۵ ولت برای بیت 1) استفاده نشده است چراکه منجر به برخی اشکالات و ابهام [در دریافت بیت‌ها] خواهد شد: مثلاً اگر ایستگاهی رشته بیت 0001000 را ارسال کند، ایستگاه‌های دیگر، ممکن است به غلط آنرا 10000000 یا01000000 تشخیص بدهد زیرا ایستگاه نمی‌تواند تفاوت بین آزاد بودن خط (معادل صفر ولت) و بیت صفر (باز هم معادل صفر ولت) را تشخیص بدهد. این مسئله را می‌توان با در نظر گرفتن ۱+ ولت برای بیت یک و ۱- ولت برای بیت صفر، حل کرد ولیکن باز هم مشکل اختلاف فرکانس نمونه‌برداری از سیگنال (نسبت به فرکانس اصلی) باقی می‌ماند. تفاوت در سرعت سیگنال ساعت فرستنده و گیرنده باعث خواهد شد که این دو از حالت سنکرون خارج شوند و در اینصورت محدودهٔ هر بیت قابل تشخیص نخواهد بود؛ بالاخص وقتی یک دنباله طولانی پی‌درپی صفر یا دنبالهٔ یک پشت سرهم ارسال شود.روش معمولی کدینگ را در شکل ۴-۱۶-الف می‌بینید.

به روشی نیاز است تا گیرنده بتواند به درستی شروع، پایان یا وسط هر بیت را بدون نیاز به یک سیگنال ساعت خارجی تشخیص بدهد. دو روش به نامهای «کدینگ منچستر» و «کدینگ منچستر تفاضلی» دارای چنین قابلیتی هستند. در روش منچستر هر بیت از لحاظ زمانی به دو نیم بیت تقسیم می‌شود: برای ارسال بیت ۱ ، در نیم بیت اوّل، ولتاژ بالا (High) و در نیمهٔ دوم ولتاژ پائین (Low) قرار داده می‌شود. برای بیت صفر برعکس عمل می‌شود: در نیمهٔ اوّل ولتاژ پائین و در نیمهٔ دوّم ولتاژ بالا ارسال می‌گردد. این روش اطمینان می‌دهد که در وسط هر بیت یک لبه (Transition) وجود دارد. اشکال روش منچستر آنست که در مقایسه با روش معمولی ، به پهنای باند دو برابر نیاز دارد چراکه طول هر پالس نصف طول یک بیت است. [در حقیقت برای هر بیت ۲ پالس ارسال می‌شود]. به عنوان مثال برای ارسال داده با سرعت ده‌مگابیت برثانیه، سیگنال تولیدی باید با نرخ ۲۰ میلیون بار در ثانیه ، تغییر سطح ولتاژ داشته باشد. روش کدینگ منچستر در شکل ۴-۱۶-ب نشان داده شده است.

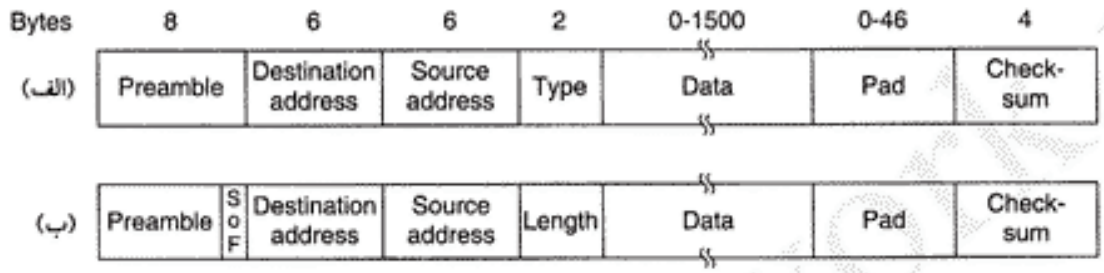
به گونه‌ای که در شکل ۴-۱۶-ج می‌بینید روش منچستر دیفرانسیلی نیز گونه‌ای از روش منچستر معمولی است. عدم وجود هرگونه لبه در ابتدای هر بیت(ابتدای Bit Time) نشان دهندهٔ بیت ۱ است و وجود لبه در ابتدای هر بیت، بیت صفر را مشخص می‌کند. در هر دو حالت ، قطعاً یک لبه در میانهٔ هر بیت وجود دارد. روش منچستر تفاضلی به ابزارهای پیچیده‌تری نیازمند است ولیکن در عوض ایمنی بیشتری در مقابل نویز از خود نشان می‌دهد. تمام سیستمهای اترنت برای سادگی از روش کدینگ منچستر معمولی استفاده کرده‌اند و در آنها سطح بالای سیگنال (حالت High) مقدار ۰/۸۵+ ولت و سطح پایین سیگنال (حالت Low) مقدار ۰/۸۵- ولت و مقدار DC سیگنال نیز صفر است. در اترنت از روش منچستر تفاضلی استفاده نمی‌شود ولیکن در برخی از شبکه‌های محلی دیگر (مثل IEEE 802.5 Token Ring) از آن استفاده شده است.



شکل ۴-۱۶. (الف) کدینگ بایتری معمولی (ب) کدینگ منچستر (ج) کدینگ منچستر تفاضلی.

٤-٣-٣ پروتکل زیرلایهٔ MAC در اترنت

قالب اصلی فریم DIX (پیشنهاد شرکت‌های DEC ، Intel ، Xerox) در شکل ۴-۱۷-الف نشان داده شده است. هر فریم با هشت بایت Preamble (دیباچه) شروع می‌شود که تمام بایتها دارای الگوی 10101010 هستند. کدینگ منچستر این الگوی هشت بایتی، بمدت ۶/۴ میکروثانیه یک سیگنال ساعت مربعی ده‌مگاهرتز تولید می‌کند تا بکمک آن گیرنده بتواند سیگنال ساعت خود را با سیگنال ساعت فرستنده سنکرون کند. گیرندگان موظفند با بهره‌گیری از ویژگی کدینگ منچستر تا انتهای فریم، سنکرون باقی بمانند و محدودهٔ بیتها را به درستی تشخیص بدهند.



شکل ۴-۱۷. قالب فریم (الف) اترنت DIX (ب) اترنت IEEE 802.3.

در این فریم دو آدرس تعریف شده است: یکی برای آدرس مقصد و یکی برای آدرس مبداء. طبق استاندارد، استفاده از آدرسهای ۲ یا ۶ بایتی مجاز است ولیکن در مشخصات معرفی شده برای استاندارد 10Mbps ، آدرسها صرفاً شش بایتی هستند.[1] اگر مقصد فریم یک ایستگاه واحد باشد، پرارزشترین بیت آدرس مقصد، صفر و اگر مقصد فریم یک گروه از ایستگاهها باشد، ۱ است. آدرسهای گروهی این امکان را فراهم می‌آورند تا چندین ایستگاه بتوانند به یک آدرس واحد گوش بدهند. وقتی فریمی به یک آدرس گروهی ارسال می‌شود تمام ایستگاه‌های گروه قادرند آنرا دریافت نمایند. ارسال فریم برای یک گروه، اصطلاحاً چندپخشی (Multicast) نامیده می‌شود. هرگاه تمام بیتهای آدرس مقصد در یک فریم، ۱ باشند به این آدرس «پخش فراگیر» (Broadcast) گفته می‌شود. فریمی که تمام بیتهای فیلد آدرس مقصد آن ۱ است توسط همهٔ ایستگاه‌های شبکه دریافت خواهد شد. تفاوت بین ارسال «چندپخشی» و «فراگیر» آنقدر مهم است که ارزش تکرار مجدد را دارد: یک فریم چندپخشی برای یک گروه انتخابی از ایستگاه‌ها در شبکهٔ اترنت ارسال می‌شود در حالیکه فریم فراگیر، برای تمام ایستگاه‌های شبکه ارسال می‌گردد. ارسال چندپخشی، انتخابی و منعطف است ولیکن نیاز به مدیریت گروه‌ها دارد. در عوض ارسال فراگیر غیرمنعطف و غیرانتخابی است ولیکن به مدیریت گروه نیازی ندارد.

یکی دیگر از ویژگیهای جالب آدرس‌دهی در اترنت آنست که بیت چهل و ششم (مجاور بیت پرارزش)، سراسری یا محلّی بودن آدرسها را مشخص می‌کند. آدرسهای محلّی، آدرسهائی هستند که توسط مسئول شبکه تعیین شده و در خارج از شبکهٔ محلی هیچ معنا و ارزشی ندارند. در مقابل آدرسهای سراسری بطور خاص و مرکزی توسط IEEE اختصاص داده می‌شوند تا این اطمینان حاصل شود که هیچ دو ایستگاهی در کلّ دنیا دارای آدرس سراسری یکسان نیستند. ۴۶ بیت باقیمانده (۴۶ = ۲ - ۴۸) فضائی معادل ۷×۱۰^{۱۳} آدرس سراسری ایجاد می‌کند. ایدهٔ اصلی آنست که هر ایستگاه به صورت یکتا و منحصربه فرد آدرس‌دهی شود. تعیین موقعیت و مشخص کردن ایستگاه مقصد، برعهدهٔ لایهٔ شبکه گذاشته شده است.

در ادامه، فیلد Type تعریف شده که به گیرندهٔ فریم تفهیم می‌کند که با این فریم چه کاری بکند. وقتی که

---

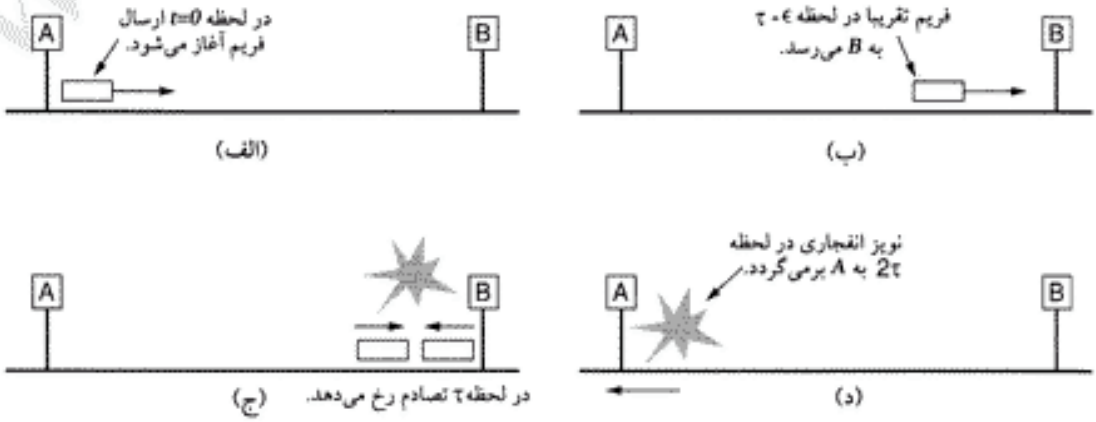
۱. امروزه در هیچیک از گونه‌های مختلف اترنت آدرسهای دوبایتی وجود خارجی ندارند. -م

چندین پروتکل لایهٔ شبکه، بطور همزمان بر روی یک ماشین واحد اجرا شده باشند، هستهٔ سیستم عامل پس از دریافت فریم، براساس این فیلد تصمیم می‌گیرد که آنرا به دست چه پروسه‌ای بسپارد. محتوای فیلد Type، پروسه‌ای که باید این فریم را تحویل بگیرد، مشخص می‌کند.

در ادامه، «فیلد داده» قرار می‌گیرد که گنجایش آن حداکثر ۱۵۰۰ بایت است. محدودیت ۱۵۰۰ بایتی، در زمان ارائهٔ استاندارد DIX ، بدلخواه انتخاب شد و انگیزه اصلی طراحان از این انتخاب، صرفه‌جویی در میزان حافظهٔ لازم برای کارتهای شبکه بوده است زیرا در آن سالها (۱۹۷۸) حافظهٔ RAM گران تمام می‌شد.

گذشته از آنکه طول فریم دارای سقف حداکثر است، حداقل طول فریم نیز محدودیت دارد. گاهی ارسال فریمهای بدون داده، مفید خواهد بود ولی استفاده از چنین فریمهایی می‌تواند مشکل‌آفرین باشد. وقتی تصادم کشف می‌شود، فرستنده از ارسال باقیماندهٔ فریم جاری صرفنظر می‌کند و بدین نحو دنباله‌ای از بیتها (که در حقیقت قطعهٔ ابتدایی فریم است) روی کانال رها می‌شود. برای تشخیص فریمهای معتبر از فریمهای آشغال ، در اترنت نیاز است که طول فریم‌های معتبر حداقل ۶۴ بایت باشد که این ۶۴ بایت از ابتدای آدرس مقصد تا انتهای فیلد Checksum در نظر گرفته می‌شود. اگر بخش دادهٔ فریم از ۴۶ بایت (۱۸-۶۴) کمتر باشد در فیلد Pad آنقدر صفر اضافه می‌شود تا اندازهٔ فریم به حداقل اندازهٔ مجاز برساند.

دلیل دیگر (و بسیار مهمتر) در محدودیت حداقل اندازه برای هر فریم، آنست که مبادا ایستگاه قبل از رسیدن اوّلین بیت از فریمش به انتهای کابل، ارسال خود را به پایان برساند زیرا ممکن است تصادم بوجود بیاید [و چون ارسال فریم خاتمه یافته ، ایستگاه از این موضوع باخبر نخواهد شد. -م] این مشکل در شکل ۴-۱۸ نشان داده شده است. در زمان t=0 ایستگاه A در یک طرف شبکه شروع به ارسال فریم روی کابل می‌کند. فرض کنید زمان تاخیر رسیدن فریم به انتهای کابل τ ثانیه باشد. دقیقاً قبل از رسیدن فریم به انتهای کابل (یعنی در لحظهٔ $\tau - \varepsilon$) آخرین ایستگاه متصل به کابل یعنی B شروع به ارسال فریم خود می‌نماید. حال وقتی B تشخیص می‌دهد که توان دریافتی از کانال از توان ارسالی بیشتر است متوجه بروز تصادم می‌شود و با قطع ارسال خود و تولید یک نویز ۴۸ بیتی قوی، بروز تصادم را به اطلاع بقیهٔ ایستگاه‌ها می‌رساند. بعبارت دیگر با ارسال یک نویز قوی این اطمینان حاصل می‌شود که هیچ ایستگاهی از پدیدهٔ تصادم بی‌خبر نخواهد ماند. در زمان حدود 2τ، فرستنده متوجه نویز شده و از ارسال دست می‌کشد. سپس قبل از تلاش مجدد برای ارسال، به اندازهٔ یک عدد تصادفی صبر می‌کند.



شکل ۴-۱۸. کشف تصادم می‌تواند تا زمان 2τ طول بکشد.

اگر ایستگاه تلاش کند فریمی کوتاه ارسال نماید و تصادم پدید بیاید، اگر چه ممکن است از تصادم مطلع شود ولی چون قبل از زمان بازگشت نویز (2τ) ، ارسال فریم به پایان رسیده ، فرستنده به غلط نتیجه می‌گیرد که فریم او

با موفقیت ارسال شده است و تکرار ارسال ممکن نخواهد بود. برای اجتناب از چنین وضعیتی ، تمام فریمها بایستی از لحاظ زمانی از (2τ) بیشتر باشند تا اطمینان حاصل شود که قبل از خاتمهٔ ارسال فریم، تصادم کشف خواهد شد. برای شبکهٔ اترنت 10Mbps با کابلی به طول حداکثر ۲۵۰۰ متر و چهار تکرارکننده (طبق تعریف 802.3) ، زمان رفت و برگشت سیگنال (با احتساب تاخیر انتشار چهار تکرارکننده) در بدترین حالت تقریباً پنجاه میکروثانیه محاسبه شده که قطعاً قابل صرفنظر نیست. بدین ترتیب حداقل طول فریم باید به اندازه‌ای باشد که از لحاظ زمانی حداقل ۵۰ میکرو ثانیه باشد. در سرعت 10Mbps هر بیت ۱۰۰ نانوثانیه طول می‌کشد لذا کوچکترین طول فریم که عملکرد صحیح شبکه را تضمین می‌کند ۵۰۰ بیت خواهد بود. برای افزودن به حاشیه امنیت این مقدار ، ۵۱۲ بیت معادل ۶۴ بایت در نظر گرفته شده است. در فیلد Pad از فریمهای کوچکتر از ۶۴ بایت ، باید آنقدر دادهٔ زائد اضافه شود تا طول آن به ۶۴ بایت برسد.

به تناسب بالا رفتن سرعت شبکه یا باید طول کانال را کاهش داد یا آنکه به طول حداقل فریمها افزود. در یک شبکهٔ LAN با کابلی به طول ۲۵۰۰ متر و نرخ ارسال 1 Gbps ، طول حداقل هر فریم، ۶۴۰۰ بایت خواهد بود! در عوض اگر بیشترین فاصلهٔ دو ایستگاه به ۲۵۰ متر کاهش یابد طول حداقل فریم به ۶۴۰ بایت کاهش خواهد یافت. این محدودیت زمانی که به سمت شبکه‌های چند گیگابیتی حرکت کنیم بسیار مشکل آفرین خواهد بود.

آخرین فیلد از فریم اترنت فیلد Checksum (کد کشف خطا) است. در حقیقت این فیلد یک کد ۳۲ بیتی استخراج شده از داده‌هاست. هرگاه تعدادی از بیتهای فریم (در اثر نویز روی کانال) اشتباه دریافت شوند، کد Chçcksum آن نادرست خواهد بود و بدین ترتیب خطا کشف می‌شود. کد کشف خطای به کار رفته در اترنت از نوع CRC است که آنرا در فصل سوم تشریح کردیم. این کدها فقط برای کشف خطا کاربرد دارند و قادر به تصحیح خطا نیستند.

زمانی که IEEE اترنت را استاندارد کرد، دو تغییر کوچک در قالب فریم DIX ایجاد کرد؛ این تغییرات در شکل ۴-۱۷-ب مشهود است. اولین تغییر آن بود که طول Preamble (دیباچه) را به ۷ بایت کاهش داد و بایت هشتم را به عنوان «بایت مشخص کنندهٔ ابتدای فریم» (Start Of Frame Delimiter) به کار برد تا با استانداردهای 802.4 و 802.5 سازگار باشد. تغییر دوّم آن بود که فیلد Type را به فیلد Length تغییر کاربری داد. البته در اینصورت گیرنده نمی‌تواند تشخیص بدهد که با فریم دریافتی چه کند ولی این مسئله با اضافه کردن یک سرآیند اضافی به قسمت داده حلّ شده است. قالب قسمت داده را در همین فصل و در توضیح «کنترل منطقی لینک» تشریح خواهیم کرد. متاسفانه در زمان ارائه 802.3 سخت‌افزار و نرم‌افزار زیادی بر اساس اترنت DIX در حال استفاده بود و شرکتهای سازنده و کاربران، چندان علاقمند به تبدیل فیلد Type به Length نبودند! در سال ۱۹۹۷، IEEE کوتاه آمد و اذعان کرد که هر دو روش قابل قبول است. خوشبختانه، تمام مقادیری که در فیلد Type درج می‌شد بالاتر از عدد ۱۵۰۰ بودند. بدین ترتیب قرار بر آن شد که اگر عدد درون این فیلد از ۱۵۰۰ بیشتر باشد به عنوان فیلد Type فرض شود و در غیر اینصورت بعنوان فیلد Length تعبیر گردد. حال دیگر IEEE می‌توانست همه را راضی نگه دارد: چه آنهائی که از استاندارد او استفاده می‌کردند و چه آنهائی که می‌خواستند کار با استاندارد DIX را بدون احساس کمبود ادامه بدهند!!

٤-٣-٤ الگوریتم عقب‌گرد نمایی

در این بخش بررسی خواهیم کرد که وقتی تصادم به وقوع می‌پیوندد روال تولید عدد تصادفی [به منظور انتظار و تلاش مجدد] چگونه است. الگوی ما کماکان مدل شکل ۴-۵می‌باشد. پس از بروز تصادم ، زمان به تعدادی برش مجزا تقسیم می‌شود؛ طول این برشها معادل با بیشترین زمان رفت و برگشت سیگنال برروی کانال (یعنی 2τ) است. با در نظر گرفتن بیشترین طول کابل در اترنت ، هر یک از این برشهای زمانی معادل ۵۱۲ بیت یعنی ۵۱/۲

میکروثانیه خواهد بود.

پس از اولین تصادم هر ایستگاه قبل از تلاش مجدد، به صورت تصادفی صفر یا یک برش زمانی منتظر می‌ماند.(یعنی بصورت تصادفی یکی از اعداد ۰ یا ۱ را تولید می‌کند.) اگر دو ایستگاه با هم تصادم کنند و اعداد تصادفی تولید شده مشابه باشند، تصادم تکرار خواهد شد. لذا در دومین تصادم متوالی، یکی از اعداد ۰، ۱، ۲، ۳ انتخاب و بهمان تعداد برش زمانی انتظار می‌کشد. [یعنی اگر عدد ۲ باشد ۵۱/۲ × ۲ میکروثانیه منتظر می‌ماند.] اگر سومین تصادم متوالی رخ بدهد (که احتمال چنین رخدادی ۰/۲۵ است) تعداد برشهای زمانی انتظار، عددی تصادفی بین صفر تا ۱ – $۲^۳$ (۷~۰) خواهد بود.

بطور عامّ در تصادم پیاپی iام، عددی تصادفی بین صفر تا ۱ – ۲^i انتخاب می‌شود و متناسب با عدد انتخابی، بر مبنای برشهای ۵۱/۲میکروثانیه‌ای منتظر می‌ماند. با این حال پس از دهمین تصادم متوالی،بازه تولید اعداد تصادفی بین صفر تا ۱۰۲۴ ثابت خواهد ماند. پس از شانزدهمین تصادم پیاپی، کنترلر کارت شبکه، دیگر ادامه نداده و پیغامی مبنی بر وجود اشکال جدّی در شبکه، به کامپیوتر گزارش خواهد کرد. تشخیص بیشتر، بر عهدهٔ لایه‌های بالاتر است.

این الگوریتم که اصطلاحاً «**عقب‌گرد نمایی دودوئی**» (Binary Exponential Backoff) نامیده می‌شود، بدین دلیل انتخاب شده تا بتواند خود را بصورت پویا با هر تعداد ایستگاه که در تلاش برای ارسال هستند، تطبیق بدهد. اگر در هر تصادم، زمان تصادفی انتظار به صورت ثابت و در محدودهٔ صفر تا ۱۰۲۳ انتخاب می‌شد اگر چه احتمال دو تصادم متوالی بسیار ناچیز بود ولی در عوض زمانی که ایستگاه‌های تصادم‌کننده باید صبر می‌کردند، بالغ بر صدها بازهٔ زمانی می‌شد و تاخیر بالا می‌رفت. برعکس، اگر بطور دائم یکی از اعداد صفر یا یک انتخاب شود آنگاه اگر به فرض صد ایستگاه سعی در ارسال کنند تصادم‌های پیاپی آنقدر تکرار می‌شود تا زمانی که ۹۹ ایستگاه تصادفاً ۱ و یکی از آنها ۰ را انتخاب کند. این اتفاق ممکن است سالها طول بکشد! اگر الگوریتمی داشته باشیم که در آن بازه‌های زمانی انتظار در پی تصادمات متوالی، بطور نمائی رشد کند، این اطمینان حاصل می‌شود وقتی تعداد ایستگاه‌های آمادهٔ ارسال کم است تاخیر کمی بوجود بیاید و وقتی تعداد ایستگاه‌های آماده ارسال زیاد است در مدت زمان معقولی مسئلهٔ تصادم حلّ شود. گذاشتن سقف ۱۰۲۳، اجازهٔ رشد بیش از اندازهٔ زمان انتظار را نخواهد داد.

بگونه‌ای که قبلاً اشاره کردیم در CSMA/CD دریافت فریم تایید نمی‌شود [یعنی پس از ارسال فریم سالم، دریافت آن گزارش نخواهد شد]. از آنجایی که جان به در بردن از تصادم تضمین‌کنندهٔ عدم خرابی بیتها در اثر نویز کانال نیست، لذا در مقصد هر فریم باید بررسی و در صورت صحّت ، پیغام تصدیق (ACK) برگردانده شود. طبعاً پیغام تایید وصول، خودش یک فریم معمولی است که برای ارسال آن نیز همانند فریم داده [برای در اختیار گرفتن کانال] باید مبارزه شود. با این وجود با یک تغییر ساده در الگوریتم رقابت، می‌توان به دریافت پیغام ACK سرعت بخشید.(Tokoro and Tamaru , 1977) تمام کاری که باید انجام شود آنست که پس از خاتمهٔ ارسال فریم، اولین بازهٔ زمانی به ایستگاه مقصد اختصاص داده شود تا پیغام ACK خود را ارسال نماید. [فریم ACK بسیار کوتاه است.] متاسفانه در استاندارد اترنت، چنین قابلیتی گنجانیده نشده است.

٤-۳-۵ کارائی (بازده) اترنت

در اینجا اجازه بدهید کارآئی اترنت را در شرایط بار سنگین و ثابت، یعنی شرایطی که در آن همیشه k ایستگاه آماده ارسال هستند، بررسی نماییم. تحلیل دقیق الگوریتم عقب‌گرد نمایی پیچیده است. به جای آن روش «متکالف» و «باگز» را دنبال کرده و فرض می‌کنیم که احتمال تکرار ارسال فریم در هر برش رقابت، ثابت باشد. اگر هر ایستگاه در خلال برش رقابت با احتمال p اقدام به ارسال فریم نماید احتمال آنکه یک ایستگاه در همان برش موفق به

ارسال شود مساویست با:

$$A = k.p.(1 - p)^{k-1}$$

A زمانی حداکثر خواهد شد که p=1/k باشد؛ با فرض p=1/k ، وقتی k به سمت بی‌نهایت میل می‌کند مقدار A ، 1/e خواهد بود. احتمال آنکه دورهٔ رقابت دقیقاً j مرحلهٔ متوالی ادامه یابد، مساوی با $A.(1-A)^{j-1}$ است فلذا میانگین تعداد دفعات تصادم در هر ارسال طبق رابطهٔ زیر بدست می‌آید:

$$\sum_{j=0}^{\infty} j.A.(1-A)^{j-1} = 1/A$$

از آنجایی که هر برش زمانی حدوداً 2τ طول می‌کشد، میانگین زمان رقابت W = 2τ/A خواهد بود. با فرض مقدار بهینه برای p (یعنی 1/e) میانگین تعداد برشهای رقابت هرگز از 2τ.e بیشتر نخواهد شد [زیرامقدار A در رابطهٔ W=2τ/A حداکثر 1/e است] بدین ترتیب W معادل 2τe یعنی حدود 5.45 خواهد بود.

با فرض آنکه تعداد ایستگاه‌های آماده ارسال ، زیاد باشد و ارسال فریم، P ثانیه به طول بینجامد داریم:

$$\text{کارآئی کانال} = \frac{P}{P + 2\tau/A}$$ رابطهٔ ۴-۶

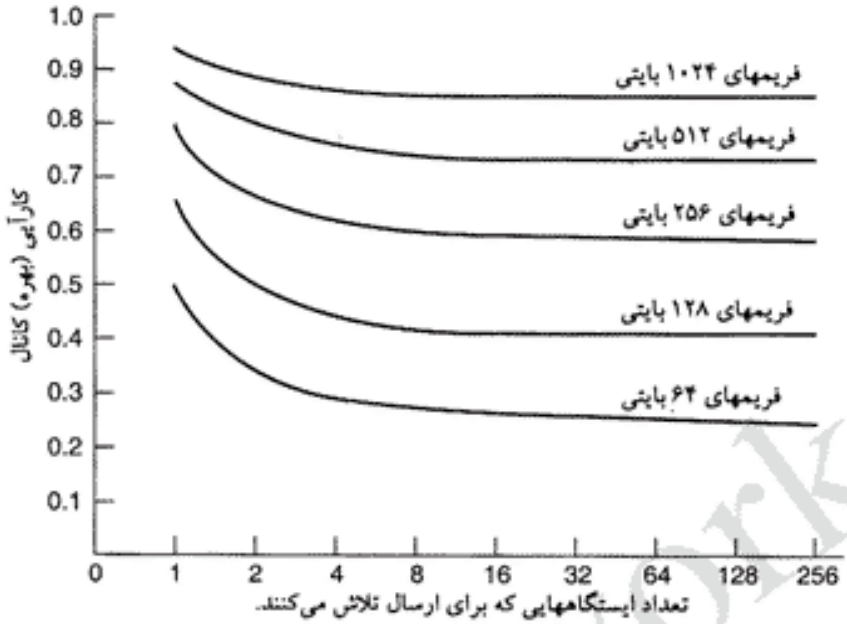
در اینجا می‌بینیم که فاصلهٔ حداکثر کابل بین دو ایستگاه در کارآئی این شبکه وارد می‌شود (زیرا مقدار τ به طول کابل بستگی دارد.). این موضوع نشان می‌دهد که برای فواصل طولانی باید به سراغ یک توپولوژی بغیر از آنچه که در شکل ۴-۱۵-الف دیده می‌شود، رفت. هر چه طول کانال زیادتر باشد طول دورهٔ رقابت بیشتر خواهد شد. این نتیجه بروشنی مشخص می‌کند که چرا در استاندارد اترنت، طول کابل محدودیت دارد. بهتر آنست که رابطهٔ ۴-۶ را به صورت واضحتر و بر حسب طول فریم ، F ، پهنای باند شبکه ، B ، طول کابل ، L و سرعت انتشار سیگنال برروی کانال ، C و مقدار بهینهٔ A یعنی 1/e بنویسیم. با قرار دادن P = F/B معادلهٔ ۴-۶ به صورت زیر در می‌آید:

$$\text{کارآئی کانال} = \frac{1}{1 + 2BLe/cF}$$ معادله ۴-۷

وقتی دومین عبارت یعنی (2B.L.e/c.F) بزرگ باشد کارآئی کانال پائین خواهد بود. بالاخص با افزایش پهنای باند یا طول کانال شبکه (یا بطور کلی با افزایش حاصل‌ضرب B.L) ، کارآئی شبکه (برای فریمهای با طول مشخص) کاهش خواهد یافت. متاسفانه بیشتر پژوهشهایی که برای بالا بردن سرعت سخت‌افزار شبکه انجام می‌شود، این حاصل‌ضرب را افزایش می‌دهد زیرا امروزه عموم مردم به پهنای باند بالا و کانالهای طولانی نیاز دارند (مثلاً برای شبکهٔ فیبر نوری MAN) و محاسبهٔ فوق نشان می‌دهد که شبکه اترنت برای چنین محیط‌هائی مناسب نیست. در ادامهٔ همین فصل خواهیم دید که روش دیگر ی برای پیاده‌سازی اترنت پیشنهاد شده که در مبحث اترنت مبتنی بر سوئیچ، بدان خواهیم پرداخت.

در شکل ۴-۱۹، منحنی کارآئی کانال بر حسب تعداد ایستگاه‌های آماده جهت ارسال و با در نظر گرفتن $2\tau = 51.2\mu s$ و نرخ ارسال B=10Mbps ترسیم شده است. [هر یک از منحنی‌ها به ازای یک F یعنی طول فریم مشخص ترسیم شده‌اند.] برای فریمهای ۶۴ بایتی کارآئی کانال چندان جالب نیست. در طرف مقابل، با نظر گرفتن ۶۴ بایت برای هر برش رقابت، میانگین زمان رقابت برای فریمهای ۱۰۲۴ بایتی ، معادل ۱۷۴ بایت بوده و راندمان کانال تقریباً ۸۵% است.

برای آنکه تعداد متوسط ایستگاه‌های آمادهٔ ارسال را در بار سنگین محاسبه نمائیم می‌توان از روش زیر بهره گرفت: هر فریم برای مدت زمانی معادل: «طول دورهٔ رقابت به اضافهٔ زمان ارسال فریم» یعنی به اندازهٔ P+w ثانیه، کانال را درگیر خواهد کرد. بدین ترتیب تعداد فریمها در هر ثانیه 1/(P+w) خواهد بود. هرگاه ایستگاه‌ها با نرخ



شکل ۴-۱۹. منحنی کارآئی کانال در اترنت 10 Mbps با فرض برشهای رقابت ۵۱۲ بیتی.

میانگین λ فریم برثانیه به تولید فریم مشغول باشند و سیستم «**در حالت k**»[1] باشد، مجموع نرخ تولید فریم در ایستگاه‌های فعال $k\lambda$ فریم برثانیه خواهد بود. از آنجایی که در حالت «**آرامش**» (Equilibrium) بایستی نرخ ورودی و خروجی یکسان باشد لذا می‌توانیم این دو عبارت را معادل هم قرار داده و آنرا بر حسب k حلّ کنیم. (دقت کنید که w خود تابعی از k است.) روش تحلیل پیچیده و دقیقتری توسط (Bertsekas and Gallager, 1992) ارائه شده است.

اشاره به این نکته خالی از لطف نیست که تاکنون حجم بسیار گسترده‌ای از مطالعات در خصوص تحلیل کارآئی شبکهٔ اترنت (و شبکه‌های دیگر) انجام شده است؛ در تمام آنها مجازاً فرض شده که ترافیک ایستگاه‌ها مبتنی بر تابع توزیع پواسون تولید می‌شود. وقتی پژوهشگران مطالعات خود را به مدل حقیقی تولید ترافیک، (در عمل) معطوف کردند، مشخص شد که ترافیک شبکه به ندرت پواسون است و «ترافیک هر ایستگاه خاصّ فقط شبیه به خودش است»! (Paxon and Floyd,1994; and Willinger et al.,1995) این بدین معناست که میانگین‌گیری در یکدورهٔ طولانی از زمان، «میانگین هموار‌شده» (Smoothed Average) ترافیک را محاسبه نخواهد کرد؛ یعنی متوسط تعداد فریمها در «هر دقیقه از ساعت» دارای واریانس (پراکندگی) بیشتری نسبت به متوسط تعداد فریمها، در «یک ثانیه از هر دقیقه» خواهد داشت. این نتیجه‌گیری بدین معناست که اغلب مدلهای عرضه شده برای ترافیک شبکه قابل اعمال در محیطهای واقعی نیست.

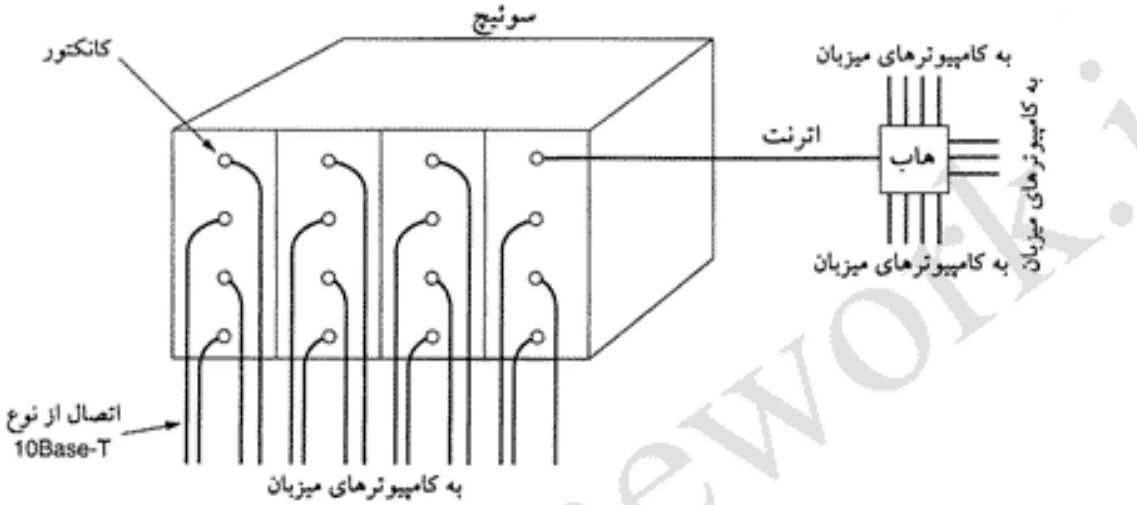
٤-٣-۶ اترنت مبتنی بر سوئیچ

هر چه بر تعداد ایستگاه‌های شبکهٔ اترنت افزوده شود، ترافیک شبکه افزایش یافته و سرانجام شبکهٔ LAN اشباع خواهد شد. یکی از روشهای کاهش این مشکل آنست که از اترنت 10Mbps به سوی اترنت 100Mbps حرکت کنیم ولیکن با رشد کاربردهای چندرسانه‌ای، اترنت 100Mbps یا حتّی 1-Gbps نیز ، اشباع خواهد شد.

---

۱. «حالت k» و «حالت آرامش» از اصطلاحات نظریهٔ صف است. برای درک این مبحث باید اندکی در خصوص نظریهٔ صف (Queuing Theory) مطالعه قبلی داشته باشید. -م

خوشبختانه راه دیگری برای مواجهه با بار سنگین وجود دارد: «اترنت مبتنی بر سوئیچ». این مدل در شکل ۴-۲۰ به تصویر کشیده شده است. در مرکز این سیستم «سوئیچ» قرار گرفته که از یک Backplane پرسرعت و فضائی برای نصب ۴ تا ۳۲ «**کارت اتصالی خط**»[1] تشکیل شده است و هر کارت ۱ تا ۸ کانکتور دارد. بطور معمول، کانکتورها از نوع 10Base-T هستند که از طریق سیمهای زوجی بهم‌تابیده، مستقیماً به یک کامپیوتر میزبان متصل می‌شوند.



شکل ۴-۲۰. مثالی ساده از اترنت مبتنی بر سوئیچ.

هرگاه ایستگاهی بخواهد یک فریم اترنت را بفرستد، آن فریم را در قالب عادّی و استاندارد سازماندهی کرده و آنرا برای سوئیچ می‌فرستد. کارت واسط دریافت‌کنندهٔ فریم، ابتدا آدرسهای آنرا بررسی می‌کند تا ببیند آیا گیرندهٔ فریم (مقصد) به همان کارت متصل است؛ اگر مقصد به همان کارت واسط متصل نبود، فریم برروی Backplane پرسرعت ارسال می‌شود تا کارت واسط مقصد آنرا دریافت کند. Backplane با سرعتی حدود چندین گیگابیت در ثانیه و با پروتکلی اختصاصی کار می‌کند.

حال ببینیم وقتی دو ایستگاه متصل به یک کارت مشابه ، به بطور همزمان فریمی را ارسال کنند چه اتفاقی می‌افتد؟ این مسئله به ساختار طراحی آن کارت بستگی دارد. امکان دارد که تمام پورتهای یک کارت از طریق یک سیم، مستقیماً به هم وصل شده و یک شبکهٔ محلی [باس] تشکیل شده باشد. آنگاه، تصادماتی که ممکن است برروی این LAN داخلی رخ بدهد بروش معمولی CSMA/CD حلّ و فصل شده و ارسال مجدّد نیز طبق الگوریتم «عقب‌گرد دودوئی نمایی» (Binary Exponential Backoff) انجام می‌گیرد. در چنین کارتهائی، از بین ایستگاه‌های متصل به هر کارت، در آن واحد فقط یک ایستگاه می‌تواند ارسال کند در حالیکه ایستگاههای متصل به کارتهای متفاوت می‌توانند بطور موازی [همزمان] ارسال داشته باشند. در این طرح هر کارت برای خودش یک «**حوزهٔ تصادم**»[2] (Collision Domain) تشکیل داده و این حوزه، مستقل از حوزهٔ تصادم دیگر کارتهاست. هرگاه فقط یک ایستگاه در هر حوزه وجود داشته باشد [یعنی فقط یک ایستگاه در هر کارت] آنگاه تصادم منتفی است و کارآئی افزایش خواهد داشت.

در انواع دیگر کارتها، هر یک از پورتهای ورودی دارای بافری اختصاصی هستند و بدین ترتیب فریمهای

---

۱. Plug-in Line Card

۲. ایستگاههایی که مستقیماً به یک کانال مشترک متصلند و برای ارسال بروش CSMA/CD رقابت می‌کنند اصطلاحاً در یک «حوزه تصادم» قرار دارند. -م

ورودی در درون حافظهٔ RAM کارت، ذخیره می‌شوند. در این ساختار تمام ایستگاه‌ها می‌توانند بطور همزمان و دوطرفه (Full Duplex)، ارسال و دریافت داشته باشند؛ عملی که در ساختار معمولی CSMA/CD ، با یک کانال واحد ممکن نیست. هنگامی که یک فریم بطور کامل دریافت شود، کارت مربوطه می‌تواند بررسی کند که آیا ماشین مقصد به پورتی برروی همان کارت متصل است یا باید به سمت پورتی بر روی کارت دیگر هدایت شود. در حالت اوّل، فریم مستقیماً به سمت مقصد ارسال می‌شود. در حالت دوّم، فریم از طریق Backplane به سمت کارت مناسب هدایت می‌گردد. در این طراحی، هر پورت برای خود «حوزهٔ تصادم» کاملاً مستقل دارد، یعنی هرگز تصادم رخ نمی‌دهد. در این ساختار، ظرفیت کلّ سیستم در مقایسه با 10Base5 (که در آن برای کل سیستم یک حوزهٔ تصادم واحد وجود دارد) چندین برابر افزایش خواهد داشت.

از آنجایی که سوئیچ، فریمهای استاندارد اترنت را از پورتهای ورودی خود می‌پذیرد لذا می‌توان از برخی پورتها در نقش «متمرکز کننده» (Concentrator) استفاده کرد. در شکل ۴-۲۰ پورت سمت چپ/بالایی سوئیچ، بطور مستقیم به یک ایستگاه واحد متصل نیست بلکه مثلاً به یک هاب ۱۲ پورت متصل است. وقتی فریمی به هاب وارد می‌شود، بروش معمول برای بدست آوردن کانال رقابت می‌کند؛ در حالیکه ممکن است تصادم رخ داده و از الگوریتم عقب‌گرد نمایی استفاده شود. فریمی که موفق به ارسال روی کانال شده و به سوئیچ وارد گردد، همانند یک فریم معمولی با آن برخورد می‌شود یعنی برای رسیدن به خط خروجی مناسب از طریق Backplane ارسال و به مقصد هدایت خواهد شد.

اگر چه هابها ارزانتر از سوئیچها هستند ولیکن با سقوط قیمت سوئیچها، هاب به سرعت از صحنه خارج می‌شود، ولی هنوز میراث هابها باقی است!

٤-۳-۷ اترنت سریع

در بدو آن ایّام، سرعت 10Mbps برای اترنت بسیار شگفت‌انگیز و رویایی به نظر می‌رسید؛ همانگونه که ظهور مودم‌های ۱۲۰۰bps در دورانی که فقط مودم‌های آکوستیکی و ابتدائی ۳۰۰bps رایج بود، پدیده‌ای اعجاب‌انگیز محسوب می‌شد؛ ولیکن هر پدیدهٔ نوظهوری به سرعت فرسوده و محو می‌شود. به عنوان نتیجهٔ قانون پارکنسیون می‌توان منتظر زمانی بود که ارسال داده‌ها با نرخی معادل با پهنای باند طبیعی هر کانال ارسال شوند. (قانون پارکینسون بیان می‌کند که: «یک کار تا زمانی که وقت برای تکمیل شدن داشته باشد، ادامه می‌یابد.») برای افزایش سرعت شبکه‌ها، گروه‌های صنعتی مختلف دو طرح جدید مبتنی برشبکهٔ حلقه (Ring) و فیبرنوری ارائه کردند:یکی از آنها طرح شبکهٔ FDDI[1] و دیگری طرح Fibre Channel[2] نام داشت. برای کوتاه کردن داستان مفصل آنها، باید اشاره کنیم که اگر چه از هر دو به عنوان ستون‌فقرات شبکه استفاده شد ولیکن هیچیک از آنها نتوانست برای اتصال کامپیوترهای رومیزی به کاربرود، چراکه در این دو شبکه، مدیریت ایستگاه‌ها بسیار پیچیده بود و نیاز به تراشه‌های پیشرفته داشت که به گران شدن قیمت آنها می‌انجامید؛ سرانجام پروندهٔ آنها به تاریخ پیوست. درسی که باید از سرنوشت اینگونه شبکه‌ها گرفت آنست که باید هر سیستمی را ساده و سهل طراحی کرد.

بهر تقدیر، ناکامی و شکست شبکه‌های LAN مبتنی بر فیبرنوری [مثل FDDI]، کمبود و خلاء اترنت سریعتر از 10Mbps را هر چه بیشتر نمایان می‌کرد. در بسیاری از محیطها، افراد به پهنای باند بیشتری نیاز داشتند و برای رفع این مشکل به ناچار چندین شبکهٔ محلی 10Mbps را نصب و از طریق تکرارکننده (Repeater) ، پل

---

۱. Fiber Distributed Data Interfce

۲. نام این طرح Fibre Channel است نه Fiber Channel، زیرا ویرایشگر این طرح بریتانیایی بوده نه آمریکایی! -اندی تانن‌بام

(Bridge) یا مسیریاب (دروازه ـ Gateway ـ) به هم متصل کرده بودند. این ساختار بهم‌پیچیده، گاهی چنان سردرگم و پراشکال می‌شد که مسئول شبکه احساس می‌کرد این شبکه‌ها به وسیلهٔ آدامس بادکنکی یا نخ به هم کوک خورده‌اند!

چنین فضا و شرایطی، IEEE را بر آن داشت تا در سال ۱۹۹۲ کمیتهٔ 802.3 را دور هم جمع کند تا در خصوص شبکهٔ محلی سریعتر از ده‌مگابیت تصمیم بگیرند. یک پیشنهاد آن بود که 802.3 بهمان شکل اصلی حفظ شده و فقط سرعت آن افزایش یابد. پیشنهاد دیگر آن بود که 802.3 بطور کلّ از نو تدوین شده و ویژگی‌های جدیدی مثل ارسال ترافیک بی‌درنگ و مبادله دیجیتالی سیگنال صوت به آن اضافه شود ولی در عین حال (بدلیل مسائل بازار و فروش) نام قبلی آن حفظ شود. پس از بحث و مناقشهٔ فراوان، کمیته تصمیم گرفت که ساختار شبکهٔ 802.3 را به همان شکل اصلی حفظ کند و فقط به سرعت آن بیفزاید.

گروهی که پیشنهاد آنها در خصوص طراحی مجدّد 802.3 پذیرفته نشد برای خودشان کمیته تشکیل دادند و شبکهٔ محلّی پیشنهادی خود را با نامی دیگر استاندارد کردند که نهایتاً شبکهٔ 802.12 پدید آمد. (کاری که معمولاً هر شخصی با دید صنعتی یا اقتصادی در چنین شرایطی انجام خواهد داد انجام طرحی است که درست و منطقی به نظر می‌رسد. ولی بازار چیز دیگری می‌گوید! تاریخ نشان داد که شبکهٔ 802.12 که شرکت هیولت پاکارد برروی آن سرمایه‌گذاری کرد اقبالی بدست نیاورد!)

به سه دلیل زیر، کمیتهٔ 802.3 تصمیم گرفت اترنت را بدون تغییر در بنیان آن، فقط توسعه بدهد:

۱. نیاز به سازگاری شبکهٔ جدید با شبکه‌های اترنت موجود

۲. نگرانی از آنکه پروتکل جدید مشکلات پیش‌بینی‌نشده داشته باشد!

۳. تمایل به آنکه قبل از تغییر تکنولوژی بتوانند کار را به اتمام برسانند و مشمول زمان نشود.

کار به سرعت انجام گرفت (البته با سرعتی که در عرف کمیتهٔ استاندارد است) و حاصل کار، استاندارد 802.3u بود که به صورت رسمی در ژوئن ۱۹۹۵ توسط IEEE تائید و معرفی شد. از دیدگاه فنی، 802.3u استاندارد جدیدی محسوب نمی‌شود بلکه یک ضمیمهٔ مکمل برای استاندارد موجود 802.3 است. (تا برسازگاری با استاندارد قبل تاکید شده باشد). از آنجایی که عموم افراد به استاندارد جدید به جای 802.3u ، «اترنت سریع» (Fast Ethernet) می‌گویند ما نیز همین اصطلاح را به کار خواهیم گرفت.

ایدهٔ اصلی در اترنت سریع بسیار ساده بود: تمام ویژگیها مثل قالب فریم ، واسطها، قواعد و الگوریتمها را بدون تغییر نگاه داشته و فقط «زمان یک بیت» را از 100 نانو ثانیه به 10 نانو ثانیه کاهش بدهیم. از دیدگاه فنی، این کار در شبکه اترنت با کابل 10Base5 یا 10Base2 بسادگی میسر است و به شرط آنکه طول حداکثر کانال با ضریب ده کاهش یابد ، تصادمها نیز بموقع کشف خواهد شد. با این وجود محاسن سیم‌کشی مبتنی بر 10Base-T (یعنی استفاده از زوج سیم‌های به هم‌تابیده) بقدری مفید و چشمگیر بود که اترنت سریع کلاً بر اساس این نوع کابل طراحی شد. لذا کلّ سیستم اترنت سریع، از هاب یا سوئیچ بهره می‌گیرد و استفاده از کابلهای چندانشعابی ، کانکتورهای BNC یا انشعاب‌های تزریقی (Vampire Tap) مجاز نیست.

ولیکن هنوز چند گزینهٔ دیگر باقی مانده بود که بایستی برای آنها نیز تصمیم گرفته می‌شد؛ مهمترین آنها، انتخاب انواع سیم زوجی بود که اترنت سریع باید از آنها پشتیبانی می‌کرد. یکی از نامزدها کابل زوجی ردهٔ ۳ (Cat 3) بود. [کابل Cat 3 برای سیم‌کشی تلفن در ساختمانها به کار می‌رود.] استدلال این انتخاب آن بود که در دنیای غرب در هر دفتر از ساختمانهای اداری حداقل چهار جفت سیم Cat 3 (یا حتی بهتر از Cat 3) از قبل وجود دارد که در فاصله‌ای حدود صد متر به جعبهٔ تقسیم تلفن کشیده شده است. بنابراین با استفاده از کابل زوجی Cat 3 این امکان وجود داشت که بدون نیاز به سیم‌کشی مجدّد ساختمان، بتوان کامپیوترهای رومیزی را از طریق اترنت

سریع به هم متصل کرد؛ این ویژگی، امتیاز بزرگی برای اغلب سازمانها و موسسات به حساب می‌آمد.
بزرگترین اشکال سیم‌های زوجی Cat 3 آنست که قادر نیستند سیگنالی با نرخ تغییر 200Mbaud/sec را در فاصلهٔ صد متر حمل کنند (100Mbps در روش منچستر معادل 200Mbaud/sec است)؛ در استاندارد 10Base-T ، فاصلهٔ صد متر ، بیشترین فاصلهٔ مجاز کامپیوتر از هاب است. از طرفی کابلهای ردهٔ ۵ (یعنی Cat 5) می‌توانند چنین سیگنالی را به راحتی در فاصلهٔ صد متری منتقل کنند؛ فیبر نوری حتی قادر است با نرخ بسیار بالاتر، این کار را انجام بدهد.

تصمیم نهائی بر آن شد که مطابق با مشخصات شکل ۴-۲۱ ، هر سه گزینه در استاندارد جدید مجاز شمرده شود، ولیکن به گزینهٔ استفاده از سیم‌های Cat 3 بهبودهائی داده شد تا بتواند سیگنال با نرخ مورد نیاز را حمل کند.

نام کابل	نوع کابل	حداکثر طول هر قطعه	مزایا
100Base-T4	Twisted pair	100 m	از کابلهای معمولی تلفن (UTP Cat 3) استفاده می کند.
100Base-TX	Twisted pair	100 m	ارسال دوطرفه کامل با نرخ 100Mbps و کابل UTP Cat 5
100Base-FX	Fiber optics	2000 m	ارسال دوطرفه کامل با نرخ 100Mbps در فواصل طولانی

شکل ۴-۲۱. کابل‌کشی اترنت سریع.

در الگوی سیم‌کشی با کابلهای معمولی Cat 3 (از نوع UTP) که 100Base-T4 نامیده می‌شود از نرخ سیگنالینگ ۲۵ مگاهرتز استفاده شده است فلذا فقط ۲۵ درصد سریعتر از استاندارد ۲۰ مگاهرتز در اترنت استاندارد است (با مرور شکل ۴-۱۶ به یاد بیاورید که در کدینگ منچستر، هر یک از ده میلیون بیت در ثانیه نیاز به دو پالس ساعت دارد.) بهر حال برای تامین پهنای باند مورد نیاز در 100Base-T4 به چهار زوج سیم Cat 3 نیاز است. [۴×۲۵MHz] از آنجایی که سالهاست در استاندارد سیم‌کشی تلفن ساختمانها، از کابلهایی استفاده می‌شود که دارای چهار جفت سیم هستند لذا بسیاری از ادارات چنین امکانی را در اختیار داشتند. البته این موضوع بدین معناست که مرکز تلفن را کنار بگذارید ولی در عوض هزینهٔ بسیار کمتری برای داشتن سیستم پست الکترونیکی سریعتر صرف خواهد شد!

از بین چهار زوج سیم، از یکی به عنوان خط ورودی دائم به هاب، از یکی به عنوان خط خروجی دائم از هاب و از دوتای دیگر بصورت قابل تنظیم در جهت فعلی ارسال [از هاب به کامپیوتر یا بالعکس] استفاده می‌شود. برای صرفه‌جویی در پهنای باند مورد نیاز، از روش کدینگ منچستر استفاده نشده است چراکه با ابداع مولدهای مدرن سیگنال ساعت و در فاصله‌ای بدین کوتاهی، اصولاً نیازی هم به کدینگ منچستر نیست. بجای آن از سیگنالهای Ternary استفاده شده که در این کدینگ، در هر سیکل سیگنال ساعت یکی از مقادیر ۰ ، ۱ یا ۲ ارسال می‌شود.(یعنی پالسها سه‌سطحی هستند.) با داشتن سه زوج سیم در جهت ارسال و روش سیگنالینگ سه‌سطحی ، ۲۷ (۳×۳×۳) سمبل مختلف قابل ارسال است؛ بدین ترتیب با ارسال هر سمبل، می‌توان ۴ بیت را (به همراه مقداری افزونگی) انتقال داد. [۴ بیت معادل ۱۶ سمبل و ۱۱ حالت افزونگی] ارسال چهار بیت در هر سیکل از سیگنالهای ۲۵ مگاهرتزی، نرخی معادل 100 مگابیت برثانیه را فراهم می‌آورد. همچنین، در جفت‌سیم باقیمانده (زوج چهارم) یک کانال معکوس ۳۳/۳ مگابیت برثانیه ایجاد می‌شود. این ساختار که اصطلاحاً 8B/6T نامیده می‌شود (بمعنای آنکه هشت بیت در چهار تریت ـTritـ نگاشته می‌شود) روشی جالب و جذابی نیست ولی بهرحال با ساختار موجود سیم‌کشی، کار می‌کند.

الگوی سیم‌کشی با سیم‌های زوجی Cat 5 (که اصطلاحاً 100Base-TX نامیده می‌شود) ساده‌تر است زیرا این سیمها قادر به حمل سیگنالهایی با نرخ ۱۲۵ مگاهرتز هستند. بدین ترتیب برای هر ایستگاه فقط به دو زوج سیم

نیاز است: یکی ورودی به هاب و دیگری خروجی از هاب. در این الگو نیز از روش کدینگ معمولی استفاده نشده است: در عوض از روش 4B/5B بهره گرفته شده که مشابه و سازگار با FDDI است. در روش 4B/5B ، هر گروه متشکل از پنج کلاک متوالی (معادل پنج بیت) و شامل ۳۲ حالت مختلف است. ۱۶ تا از این ترکیبات برای ارسال ۴ بیت داده به کار می‌رود. [بعبارت ساده‌تر در روش 4B/5B به ازای هر چهار بیت، پنج بیت ارسال می‌شود.] برخی از ۱۶ حالت باقیمانده برای عملیات کنترلی نظیر مشخص کردن ابتدا و انتهای فریم کاربرد دارد. ترکیبات شانزده‌گانهٔ داده به نحوی انتخاب شده‌اند که در الگوی سیگنال تولیدی، لبهٔ لازم برای سنکرون ماندن سیگنال ساعت وجود داشته باشد. سیستم 100Base-TX دو طرفهٔ همزمان است: ایستگاه می‌تواند داده‌ها را با نرخ ۱۰۰ مگابیت در ثانیه ارسال و بطور همزمان دریافت نماید. اغلب به روشهای 100Base-TX و 100Base-T4 به اختصار 100Base-T گفته می‌شود.

آخرین گزینه، 100Base-FX است که در آن از دو رشتهٔ فیبرنوری مالتی‌مود (Multimode) استفاده می‌شود؛ هر یک از تارهای نوری قادر به حمل ۱۰۰ مگابیت در ثانیه به صورت همزمان هستند. مضاف بر این، فاصلهٔ بین ایستگاه و هاب تا ۲ کیلومتر قابل افزایش است.

در پاسخ به نیاز عمومی، کمیتهٔ IEEE 802 در سال ۱۹۹۷ روش کابل‌کشی جدیدی به نام 100Base-T2 به استاندارد اضافه کرد که اجازه می‌داد اترنت سریع با دو زوج سیم معمولی Cat 3 کار کند ولیکن بدلیل استفاده از روش کدینگ خاصّ ، به یک پردازندهٔ سیگنال دیجیتال (DSP Processor) پیچیده نیاز است که این انتخاب را اندکی گران قیمت می‌کند. اکنون از این ساختار به دلائلی مثل پیچیدگی و قیمت بالا و این حقیقت که بسیاری از ساختمان‌های اداری به سیم‌کشی با کابل Cat 5 تن داده‌اند ، به ندرت استفاده می‌شود.

در 100Base-T ، جهت اتصال ایستگاه‌ها به یکدیگر می‌توان از دو نوع ابزار استفاده کرد: هاب یا سوئیچ. (شکل ۴-۲۰) در هاب تمام خطوط ورودی (یا حداقل تمام ورودی‌هائی که به یک کارت واحد وارد می‌شوند) به صورت منطقی به هم متصل هستند و یک حوزهٔ تصادم واحد را ایجاد می‌کنند. در این ابزار دقیقاً مثل اترنت، تمام قواعد استاندارد (نظیر الگوریتم عقب‌گرد نمایی) اعمال می‌شود، خصوصاً آنکه در هر لحظه تنها یک ایستگاه می‌تواند ارسال داشته باشد؛ بعبارت دیگر، در هاب ماهیت ارتباط ، «دوطرفهٔ غیرهمزمان» (Half Duplex) است.

در یک سوئیچ هر یک از فریمهای ورودی برروی حافظهٔ کارت متصل به خط ، بافر می‌شود و در صورت نیاز از طریق یک Backplane بسیار سریع، از کارت مبداء به کارت مقصد هدایت می‌گردد. ساختار Backplane استانداردسازی نشده و نیازی هم بدین امر نبوده است چرا که به عنوان بخشی پنهان در درون سوئیچ انجام وظیفه می‌کند. با تکیه بر تجارب گذشته می‌توان پیش‌بینی کرد که تولیدکنندگان سوئیچ به شدت در حال رقابت هستند تا Backplane سوئیچها سریعتر شده و کارآئی کلّ سیستم افزایش یابد. از آنجایی که کابلهای 100Base-FX برای کشف بموقع تصادم در شبکهٔ اترنت بیش از اندازه طولانی هستند، فلذا این کابلها لزوماً بایا. به سوئیچها متصل شوند؛ بدین ترتیب هر ایستگاه برای خود یک «حوزهٔ تصادم» مستقل پدید آورده و تصادم منتفی خواهد بود. استفاده از هاب در 100Base-FX مجاز نیست.

به عنوان آخرین نکته باید اشاره کرد که تمام سوئیچها (مجازاً) می‌توانند با تلفیقی از ایستگاه‌های 10Mbps و 100Mbps کار کنند تا ارتقاء سیستمهای موجود به سیستمهای سریعتر ساده‌تر شود. در صورت افزایش نیاز یک سایت به تعداد بیشتری ایستگاه 100Mbps ، تنها کافی است به تعداد لازم کارت‌های ورودی خریداری شده و درون سوئیچ نصب شود. [به شکل ۴-۲۰ نگاه کنید.] در حقیقت در خود استاندارد روشی پیش‌بینی شده تا دو ایستگاه بتوانند بر سر نرخ ارسال (10Mbps یا 100Mbps) با هم توافق کرده و ماهیت ارتباط (Full Duplex یا Half Duplex بودن ارتباط) را انتخاب نمایند. بسیاری از محصولات اترنت از این ویژگی برخوردار هستند تا

بتوانند به صورت خودکار خودشان را پیکربندی نمایند.

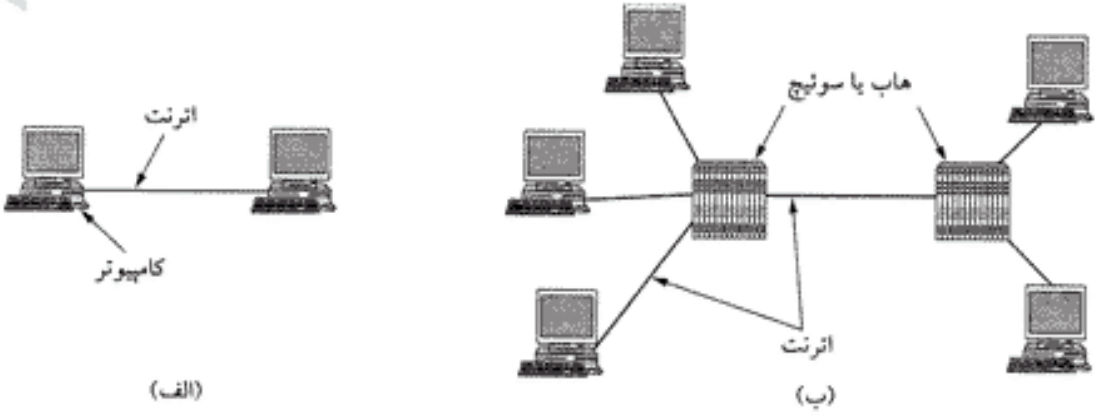
٤ـ۳ـ۸ اترنت گیگابیت (Gigabit Ethernet)

هنوز جوهر مستندات استاندارد اترنت سریع، خشک نشده بود که کمیتهٔ 802 کار را برروی اترنت سریعتر از آن آغاز کرد (۱۹۹۵). این تحقیقات سریعاً تحت عنوان اترنت گیگابیت به ثمر رسید و در سال ۱۹۹۸ توسط IEEE با نام 802.3z تصویب شد. پیشنهاد حرف z در نام این استاندارد بدین مضمون بود که اترنت گیگابیت آخرین حلقه در مسیر تکاملی اترنت خواهد بود مگر آنکه کسی بتواند حرف جدیدی پس از z ابداع کند!! در ادامه برخی از ویژگی‌های اساسی اترنت گیگابیت را تشریح خواهیم کرد. برای آگاهی بیشتر می‌توان به مقالهٔ (Seifert, 1998) مراجعه کرد.

اهداف کمیتهٔ 802.3z مشابه با اهداف کمیتهٔ 802.3u بود: اترنت ده برابر سریعتر شود و همچنان با استاندارد موجود اترنت سازگاری داشته باشد. بویژه اترنت گیگابیت باید از خدمات «انتقال دیتاگرام بدون تصدیق دریافت فریم»[1] به صورت تک‌پخشی و چندپخشی[2] و با استفاده از همان ساختار ۴۸ بیتی آدرس موجود، پشتیبانی می‌کرد. همچنین قالب فریمها و طول حداقل و حداکثر هر فریم باید مشابه با قبل انتخاب می‌شد. استاندارد نهائی، تمام این اهداف را برآورده کرده است.

کلّ پیکربندی شبکهٔ اترنت گیگابیت، به جای آنکه همانند اترنت کلاسیک دارای ساختار باس چندااتصالی (Multidrop) باشد صرفاً نقطه‌به‌نقطه (Point To Point) است. [یعنی به جای آنکه همهٔ ایستگاه‌ها به یک کابل مشترک متصل باشند بطور مستقیم به یک سوئیچ متصل می‌شوند. -م] ساده‌ترین الگوی پیکربندی اترنت گیگابیت، در شکل ۴-۲۲-الف نشان داده است؛ در این شکل دو کامپیوتر به صورت مستقیم به هم متصل شده‌اند. الگوی رایج دیگر که در شکل ۴-۲۲-ب دیده می‌شود شامل یک سوئیچ یا هاب است که به چندین کامپیوتر یا چند سوئیچ یا هاب دیگر متصل هستند. در هر دو الگو، به هر کابل اترنت صرفاً دو ابراز متصل است، نه کمتر نه بیشتر [یعنی دو کارت اترنت گیگابیت در دو سرکابل].

اترنت گیگابیت می‌تواند به دو روش عمل کند: «حالت دوطرفهٔ همزمان» (Full Duplex) و «حالت دوطرفهٔ غیرهمزمان» (Half Duplex). عملکرد طبیعی و پیش‌فرض سیستم، حالت دو طرفهٔ همزمان است و بدین ترتیب این امکان وجود دارد که ترافیک داده‌ها بتواند به صورت همزمان در دو جهت جریان داشته باشد. از این حالت زمانی استفاده می‌شود که یک سوئیچ مرکزی وجود داشته و به کامپیوتر دیگر (یا سوئیچهای دیگر) در پیرامون



شکل ۴ـ۲۲. (الف) اترنت گیگابیت با دو ایستگاه. (ب) اترنت گیگابیت با چند ایستگاه.

۱. Unacknowledged Datagram ۲. Unicast & Multicast

خود متصل باشد. در این پیکربندی تمام خطوط بافر شده‌اند[1] و طبعاً هر کامپیوتر یا سوئیچ اجازه دارد هر زمان که‌تمایل داشت فریم خود را ارسال کند. فرستند مجبور نیست کانال را شنود کند و حضور دیگران برروی کانال راتشخیص بدهد چرا که در این ساختار اصولاً هرگونه رقابتی منتفی است و هرگز تصادم رخ نخواهد داد. از آنجایی که بین سوئیچ و کامپیوتر دو کابل مستقل وجود دارد، ارسال همزمان از کامپیوتر به سوئیچ (حتی اگر سوئیچ در حال ارسال فریم برای همان کامپیوتر باشد)، میسّر است. از آنجایی که هیچ رقابتی مطرح نیست لذا پروتکل CSMA/CD کاربردی ندارد و حداکثر طول کانال فقط بر اساس توان سیگنال ارسالی [و میزان تضعیف آن روی کابل] تعیین می‌شود و ربطی به مدت زمان بازگشت نویز حاصل از تصادم [و تاخیر انتشار کابل] ندارد. سوئیچها آزادند که با نرخ مختلف ارسال و به صورت تطبیقی عمل کنند. در اترنت گیگابیت همانند اترنت سریع، از پیکربندی خودکار حمایت می‌شود.

حالت دیگر عملکرد اترنت گیگابیت، حالت دو طرفه غیر همزمان (Half Duplex) است و زمانی کاربرد دارد که کامپیوترها به جای وصل به سوئیچ به هاب متصل باشند. یک هاب فریمهای ورودی را بافر نخواهد کرد و در عوض به صورت داخلی تمام خطوط را به صورت الکتریکی به هم متصل کرده و همانند اترنت کلاسیک کابلی چنداتصالی را شبیه‌سازی می‌کند. در این حالت وقوع تصادم محتمل بوده و به پروتکل CSMA/CD نیاز خواهد بود. از آنجایی که کوتاه‌ترین فریم مجاز (یعنی فریم ۶۴ بایتی)، در اترنت گیگابیت صدبرابر سریعتر از اترنت کلاسیک ارسال می‌شود لذا حداکثر طول کابل بایستی ۱۰۰ برابر کاهش یابد تا آنکه ویژگی بنیادی مورد نیاز در CSMA/CD یعنی «عدم خاتمهٔ ارسال فریم قبل از بازگشت نویز حاصل از تصادم»، برآورده شود.(یعنی طول کابل باید حداکثر ۲۵ متر باشد.) باکابلی به طول ۲۵۰۰ متر و سرعت 1Gbps ، قبل از آنکه یک فریم ۶۴بایتی بتواند حتی یکدهم از مسیر خود را طی کند، فرستندهٔ آن ارسال خود رابه پایان رسانده و از تصادم احتمالی مطلع نخواهد شد!

کمیتهٔ 802.3z بدین نتیجه رسید که شعاع ۲۵ متر برای اترنت گیگابیت قابل قبول و مناسب نیست، فلذا دو ویژگی جدید به استاندارد افزود تا شعاع شبکه افزایش یابد. اوّلین ویژگی که «**توسیع حامل**» (Carrier Extension) نامیده شده سخت‌افزار را وادار می‌کند تا آنقدر اطلاعات زائد در انتهای فریم معمولی اضافه کند تا طول فریم حداقل ۵۱۲ بایت شود. از آنجایی که این اطلاعات زائد در سخت‌افزار مبداء اضافه و در سخت‌افزار مقصد حذف می‌شوند لذا نرم‌افزار از انجام چنین عملی بی‌اطلاع خواهد بود و هیچ تغییری در ساختار نرم‌افزار موجود نیاز نخواهد بود. البته طبیعی است که فرستادن فریمی ۵۱۲بایتی برای ارسال ۴۶ بایت دادهٔ خام کاربر، بهرهٔ مفیدی معادل ۹ درصد دارد.

دوّمین ویژگی که اصطلاحاً Frame Bursting نامیده شده اجازه می‌دهد که فرستنده دنباله‌ای از چندین فریم را در یک بار ارسال بفرستد. اگر مجموع دنبالهٔ فریمها کمتر از ۵۱۲بایت شود باز هم سخت‌افزار، داده‌های زائد [به انتهای آخرین فریم] اضافه می‌کند. هرگاه تعداد فریمهای منتظر ارسال کافی باشد این روش بسیار کارآمد و نسبت به روش «توسیع حامل» ارجح‌تر است. این ویژگی جدید، شعاع شبکه را به ۲۰۰ متر افزایش می‌دهد که برای بسیاری ادارات و موسسات کافی است. اگرچه در اترنت گیگابیت از CSMA/CD پشتیبانی می‌شود ولیکن تصور آن بسیار دشوار است که سازمانی کارتهای اترنت گیگابیت خریداری و نصب کند ولی آنها را به یک هاب متصل و اترنت معمولی را شبیه‌سازی نماید! [بهرهٔ واقعی اترنت گیگابیت با سوئیچها بدست می‌آید نه با هاب]

اگر چه هابها از سوئیچها بسیار ارزانتر هستند ولی هنوز قیمت کارت‌های اترنت گیگابیت نسبتاً گران است.

---

۱. یعنی داده‌های ارسالی مستقیماً به درون حافظه موقت منتقل می‌شوند. ـم

خرید هابهای ارزان قیمت [برای کارتهای گرانقیمت گیگابیتی] و کاستن از بهرهٔ واقعی سیستم، کوته‌فکری به نظر می‌رسد! بهرحال ویژگی «سازگاری با قبل» یکی از نگرانیهای صنایع کامپیوتریست و طبعاً نیاز بوده که کمیتهٔ 802.3z آنرا در استاندارد خود لحاظ کند. [ولیکن دلیلی ندارد از ویژگی‌های جدید آن به بهای سازگاری با قبل صرفنظر شود]

اترنت گیگابیت مطابق با فهرست ۴-۲۳ از کابلهای مسی و فیبرهای نوری پیشتیبانی می‌کند. تولید سیگنال با نرخی نزدیک به 1-Gbps بدان معناست که منبع مولد نور باید در زمانی زیر یک نانوثانیه خاموش و روشن شود. مولد نور معمولی یعنی LED نمی‌تواند با این سرعت عمل کند و طبعاً به لیزر نیاز است. استفاده از پرتوهای لیزر با طول موج ۰/۸۵ میکرون (طول موج کوتاه) و ۱/۳ میکرون (طول موج بلند) مجاز است. لیزرهای 0.85 میکرونی ارزانتر هستند ولیکن برروی فیبرنوری تک‌مود (Single Mode) کار نمی‌کنند.

نام کابل	نوع کابل	حداکثر طول هر قطعه	مزایا
1000Base-SX	Fiber optics	550 m	از فیبرهای چندموده ۵۰ و ۶۲/۵ میکرون استفاده می کند.
1000Base-LX	Fiber optics	5000 m	با فیبرهای تک‌موده ۱۰ میکرون یا چندموده ۵۰ و ۶۲/۵ میکرون
1000Base-CX	2 Pairs of STP	25 m	از کابلهای زوجی زره‌دار (STP) استفاده می کند.
1000Base-T	4 Pairs of UTP	100 m	از کابلهای استاندارد UTP Cat 5 استفاده می کند.

شکل ۴-۲۳. کابل‌کشی اترنت گیگابیت.

در اترنت گیگابیت ، استفاده از فیبرهای نوری با قطر ۱۰ ، ۵۰ و ۶۲/۵ میکرون مجاز است. مورد اوّل برای فیبرنوری تک‌مود(Single Mode) و دو مورد بعدی برای فیبرهای چندموده هستند. تمام شش ترکیب مختلف [یعنی ترکیبات مختلف دو نوع لیزر و سه قطر] مجاز نیست ولیکن طول حداکثر کابل به ترکیب مورد استفاده وابسته است. اعدادی که در شکل ۴-۲۳ ارائه شده‌اند برای بهترین حالت ممکن هستند. رسیدن به طول کابل ۵۰۰۰ متری فقط با لیزر ۱/۳ میکرون و صرفاً با کابل تک موده به قطر ۱۰ میکرون امکان‌پذیر است ولیکن این انتخاب علیرغم قیمت گران آن بهترین گزینه برای پیاده سازی ستون فقرات شبکه‌های ناحیه‌ای (Campus) محسوب می‌شود.

در الگوی 1000Base-CX ، از کابلهای مسی زره‌دار کوتاه (STP) استفاده شده است. این انتخاب از یک طرف با فیبرنوری با کارآئی بالا و از طرف دیگر با سیم‌های ارزان‌قیمت UTP در رقابت است زیرا نه به ارزانی UTP است و نه به کارآمدی فیبر نوری! لذا احتمالاً از آن استقبالی نخواهد شد.

آخرین انتخاب، استفاده از چهار زوج سیم Cat 5 است که همزمان با یکدیگر کار می‌کنند. بدلیل آنکه حجم زیادی از این نوع کابل از قبل نصب شده لذا می‌توان از آن به عنوان گونهٔ فقیرانه و کم خرج اترنت گیگابیت یاد کرد!

اترنت گیگابیت برای کدینگ سیگنال روی فیبرنوری از روشی جدید استفاده می‌کند. بهره‌گیری از روش منچستر با سرعت 1-Gbps ، نیاز به تغییر در سطح سیگنال با نرخی معادل 2-G baud/sec خواهد داشت که رسیدن به آن بسیار دشوار بوده و نیاز به پهنای باند بسیار زیادی دارد. در عوض از روشی جدید به نام 8B/10B استفاده شده است. در این روش هر بایت هشت بیتی قبل از ارسال برروی فیبرنوری به یک الگوی ده بیتی نگاشته می‌شود؛ به همین دلیل نام آن 8B/10B انتخاب شده است. از آنجایی که کلمهٔ کد ده بیتی خروجی (که به ازای هر بایت ورودی تولید می‌شود) دارای ۱۰۲۴ حالت مختلف است لذا برای هر یک از ۲۵۶ حالت مختلف ورودی می‌توان یک کلمهٔ کد، متناسب با شرایط کانال انتخاب کرد. برای انتخاب کلمهٔ کد، دو قاعدهٔ زیر به کارگرفته می‌شود:

۱. هیچ کلمهٔ کدی نباید بیش از ۴ بیت مشابه و پشت سرهم داشته باشد.

۲. هیچ کلمهٔ کدی نباید جمعاً بیش از ۶ بیت صفر یا ۶ بیت یک داشته باشد.

این معیارها باعث خواهد شد که در جریان سیگنال تولید شده در خروجی بقدر کافی لبه (Transition) وجود داشته باشد تا این اطمینان حاصل شود که گیرنده با فرستنده هماهنگ و سنکرون باقی مانده و تعداد صفرها و یکهای ارسالی بر روی فیبر، حتی‌الامکان مساوی یا نزدیک به هم باشد. مضاف بر این بسیاری از بایت‌های ورودی می‌توانند بیش از یک کلمهٔ کد هم‌ارز داشته باشند. وقتی که کدکننده، در انتخاب کلمهٔ کد آزادی انتخاب داشته باشد می‌تواند کلمهٔ کدی را انتخاب کند که برآیند تعداد صفرها و یکهائی که تا کنون ارسال شده‌اند حتی‌الامکان معادل باشد. تاکیدی که برروی معادل بودن تعداد صفرها و یک‌ها وجود دارد از آن جهت است که مولفهٔ DC سیگنال حتی‌الامکان پائین بوده و در صورت عبور از مبدلها(Transformers) تغییری در شکل سیگنال ایجاد نشود. اگر چه دانشمندان کامپیوتر تمایلی ندارند که خصوصیات یک ترانسفورمر، نوع کدینگ آنها را تعیین کند ولیکن بهرحال نکته‌ای است که باید رعایت شود.

در اترنت گیگابیت برای الگوی سیم‌کشی 1000Base-T ، از روش کدینگ متفاوتی بهره گرفته شده است چراکه ارسال پالسهای داده بر روی سیم مسی در زمان یک نانو ثانیه بسیار دشوار است. در کدینگ جدید از چهار زوج سیم Cat 5 استفاده شده تا چهار سمبل به صورت همزمان و موازی ارسال شوند. هر سمبول با پنج سطح ولتاژ متفاوت کد می‌شود. این الگو اجازه می‌دهد تا یک سمبول بتواند یکی از حالات 00 ، 01 ، 10 ، یا 11 و یک حالت کنترل خاص را کد نماید. بنابراین در هر سیکل سیگنال ساعت، برروی هر زوج سیم ۲ بیت و برروی کل سیمها ۸بیت ارسال می‌شود. سیگنال ساعت در فرکانس ۱۲۵ مگاهرتز کار می‌کند که اجازهٔ ارسال یک گیگابیت در هر ثانیه را خواهد داد. دلیل آنکه بجای استفاده از چهار سطح ولتاژ از پنج سطح استفاده شده آنست که برخی از ترکیبات آن برای عملیات کنترلی و فریمینگ باقی بماند.

سرعت 1 Gbps بسیار سریع و بالاست: به عنوان مثال هرگاه یک گیرنده، فقط برای یک میلی‌ثانیه به کاری دیگر مشغول باشد و نتواند بافر ورودی یکی از خطوط را خالی کند، در این وقفهٔ یک‌میلی‌ثانیه‌ای، ۱۹۵۳ فریم در بافر جمع خواهد شد!! همچنین اگر کامپیوتری در شبکهٔ اترنت گیگابیت، داده‌های را برای کامپیوتری در شبکهٔ اترنت کلاسیک بفرستد به احتمال بسیار زیاد داده‌ها در بافر روی هم نوشته شده و از دست خواهد رفت. پیامد این دو پدیده آن بود که در اترنت گیگابیت از کنترل جریان (Flow Control) پشتیبانی شود. (همچنان که در اترنت سریع نیز کنترل جریان وجود دارد ولیکن بروشی متفاوت)

برای عملیات کنترل جریان یکی از طرفین با ارسال یک فریم کنترلی خاص ، از طرف مقابل خود می‌خواهد که برای مدتی ارسال داده را متوقف نماید. فریمهای کنترلی، فریمهای معمولی اترنت هستند که در فیلد Type آنها عدد 0x8808 قرار گرفته است. در این صورت، دو بایت ابتدائی از فیلد داده نوع فرمان را مشخص می‌کند و بایت‌های بعدی به عنوان پارامتر آن فرمان تلقی می‌شوند (در صورت وجود). برای کنترل جریان، فریم PAUSE به کار می‌رود و پارامتر آن مدت زمان توقف را (بر مبنای طول حداقل فریم) مشخص می‌نماید. در اترنت گیگابیت واحد زمان ۵۱۲ نانو ثانیه است و فریم PAUSE می‌تواند ایستگاه را تا ۳۳/۶میلی‌ثانیه متوقف کند.

پس از آنکه اترنت گیگابیت استاندارد شد کمیتهٔ ۸۰۲ که خسته شده بود می‌خواست پی کار خود برود ولی IEEE از آنها خواست که کار را برروی اترنت ده گیگابیت شروع کنند. پس از جستجوی سخت برای حرفی که بتواند جایگزین z [در 802.3z] شود آنها به این نتیجه رسیدند که از پسوند دو حرفی استفاده کنند!! کار آنها نتیجه داد و استاندارد مربوطه با نام 802.3ae به تایید IEEE رسید. آیا اترنت صد گیگابیت بر ثانیه نیز می‌تواند محقق شود؟

٤ـ۳ـ۹ IEEE 802.2: کنترل منطقی لینک

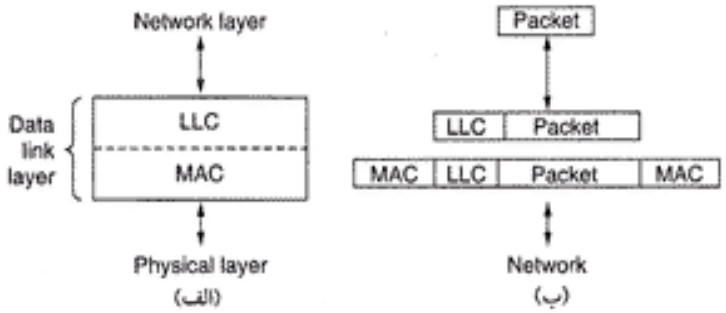
شاید زمان آن فرا رسیده باشد تا اندکی به عقب بازگردیم و برآنچه که در این فصل و ماقبل آن آموختیم، مروری تطبیقی و مقایسه‌ای داشته باشیم. در فصل ۳ آموختیم که چگونه دو ماشین می‌توانند با استفاده از پروتکلهای پیوند داده، بر روی خطی غیرقابل‌اعتماد، به صورت مطمئن مبادلهٔ اطلاعات کنند. این پروتکلها امکان نظارت برخطا (با استفاده از تصدیق دریافت فریم ـ Ack ـ) و کنترل جریان (با استفاده از پنجرهٔ لغزان) را فراهم آورده‌اند.

برعکس، در این فصل در خصوص مبادلهٔ مطمئن داده‌ها سخنی به میان نیاوردیم. تمام‌گونه‌های اترنت و دیگر پروتکلهای ۸۰۲ تلاش می‌کنند سرویس دیتاگرام عرضه کنند. [یعنی ارسال داده‌ها بدون تصدیق دریافت آنها ـAck ـ]. در اغلب موارد، این سرویس کافی به نظر می‌رسد. به عنوان مثال برای انتقال بسته‌های IP، به هیچ تضمینی در رسیدن بسته‌ها نیازی نیست. یک بسته IP را می‌توان بسادگی درون فیلد حمل داده از فریم ۸۰۲ قرار داد و آنرا ارسال کرد. اگر فریم از بین برود مهم نیست.

علیرغم این، سیستمهائی وجود دارند که به پروتکلی با قابلیتهای نظارت برخطا و کنترل جریان نیاز دارند. IEEE پروتکلی تعریف کرده که می‌تواند بر روی پروتکل اترنت یا هر پروتکل سری ۸۰۲ قرار گرفته و اجرا شود. این پروتکل که LLC (Logical Link Control) نامیده می‌شود، قادر است تفاوتهای انواع مختلف شبکه‌های ۸۰۲ را از طریق تعریف یک قالب و واسط (Interface) واحد و مشترک مخفی کند. این پروتکل بسیار شبیه به پروتکل HDLC است که در فصل ۳ آنرا بررسی کردیم. LLC نیمهٔ بالائی لایهٔ پیوند داده‌ها را تشکیل می‌دهد، در حالیکه زیرلایهٔ MAC، نیمهٔ پائینی این لایه محسوب می‌شود. (شکل ۴-۲۴)

استفادهٔ رایج از LLC بدین ترتیب است: لایهٔ شبکه در ماشین فرستنده، بسته‌ای را بکمک توابع پایهٔ و بنیادی لایهٔ LLC، بدان تحویل می‌دهد. زیرلایهٔ LLC سرآیند لازم را به بسته می‌افزاید؛ این سرآیند شامل شمارهٔ ترتیب (Seq No) و شمارهٔ تصدیق (Ack No) است. بستهٔ حاصل، درون فیلد حمل داده از فریم ۸۰۲ قرار گرفته و ارسال می‌شود. در گیرنده نیز عکس این فرآیند انجام می‌شود.

LLC از سه ردهٔ خدمات حمایت می‌کند: (۱) خدمات ارسال نامطمئن دیتاگرام، (۲) خدمات دیتاگرام باتصدیق وصول و (۳) خدمات ارسال اتصال‌گرای مطمئن. سرآیند LLC سه فیلد را در برمی‌گیرد: «نقطه دسترسی در مقصد»، «نقطهٔ دسترسی در مبداء» و «فیلد کنترل». نقطهٔ دسترسی مشخص می‌کند که این فریم از چه پروسه‌ای آمده و به کدام پروسه باید تحویل شود که در حقیقت نقش همان فیلد Type در فریم DIX را ایفاء می‌کند. فیلد کنترل نیز در برگیرندهٔ شمارهٔ ترتیب و شمارهٔ تصدیق است که شباهت فراوانی با ساختار HDLC دارد (شکل ۳-۲۴)، ولی کاملاً با آن یکسان نیست. از این فیلدها زمانی استفاده می‌شود که در سطح لایهٔ پیوند داده‌ها به یک اتصال مطمئن (Reliable Connection) نیاز باشد که در این حالت شبیه به روشی عمل می‌شود که در فصل سوّم تشریح شد. برای شبکهٔ اینترنت، صرف تلاش در تحویل بستهٔ IP کفایت می‌کند و به هیچ پیغام اعلام وصول فریم در سطح LLC نیازی نیست.



شکل ۴-۲۴. (الف) موقعیت LLC در پشتهٔ پروتکلی (ب) قالبهای پروتکل.

٤-٣-١٠ نگاهی به گذشتهٔ اترنت

اترنت برای حدود ۲۰ سال در صحنه بوده و تقریباً هیچ رقیب جدّی نداشته است و به نظر می‌رسد در سالهای آتی نیز همچنان یکّه‌تاز باشد. تعداد بسیار کمی سیستم عامل، زبان برنامه نویسی یا معماری CPU وجود داشته که بتواند برای دو دههٔ متوالی برقلهٔ افتخار بایستد و در حال ورود به دههٔ سوّم افتخار خود باشد. روشن است که اترنت حرفهائی برای گفتن داشته که بدین گونه دوام آورده است؛ اینها چه بوده‌اند؟

شاید دلیل اصلی بقای اترنت «سادگی» و «قابلیت اعتماد» آن بوده است. در عمل، معیار «سادگی» به «قابلیت اعتماد»، «ارزانی»، «سهولت نصب و نگهداری» تعبیر می‌شود. وقتی در اترنت انشعابات تزریقی (Vampire Tap) با کانکتورهای BNC عوض شد خرابیها به شدت کاهش یافت. بطور معمول، عموم افراد حاضر نیستند از ابزارهایی که بخوبی کار می‌کنند رو بگردانند و این پافشاری از آنجاست که بر همه روشن شده بسیاری از محصولات بنجل در صنعت کامپیوتر باهیاهو می‌آیند ولی بسیار ضعیف عمل می‌کنند، حتی گاهی محصولاتی که با عنوان «ارتقاء» (Upgrade) معرفی می‌شوند بسیار بدتر و ناسازگارتر از وقتی عمل می‌کنند که بطور کامل عوض شوند! ولیکن نسخه‌های ارتقایافتهٔ اترنت، اعتماد عمومی را جلب کرد.

«سادگی» [در مورد اترنت] به ارزان بودن نیز تعبیر می‌شود. اترنت نازک و سیم‌های زوجی نسبتاً کم بها و ارزان هستند. کارت‌های واسط اترنت نیز بسیار ارزانند. فقط در دورانی که هاب و سوئیچ معرفی شد، به مقداری سرمایه‌گذاری نیاز داشت ولیکن در زمان عرضهٔ آنها، شبکهٔ اترنت بقدر کافی جا افتاده بود.

اترنت در لحاظ نصب و نگهداری ساده است. هیچ نرم‌افزار اضافه‌ای نباید نصب شود (مگر نرم افزارهای بسیار کوچک راه‌انداز آنها) و به هیچ جدول یا تنظیمات پیکربندی خاصی نیاز ندارد که مدیریت آن (یا هرگونه اشتباه در تنظیم آن) کار را دشوار کند. در ضمن اضافه کردن یک ماشین جدید به شبکه به سادگی وصل آن به هاب یا سوئیچ است و کار چندانی ندارد.

امتیاز دیگر اترنت آن بود که بسادگی با TCP/IP که پروتکل غالب دنیاست، کار می‌کند. IP پروتکلی بدون‌اتصال (Connectionless) و دقیقاً متناسب با اترنت است که آن هم بدون اتصال عمل می‌کند. در مقابل ، IP سازگاری بسیار کمی با ATM (که اتصال‌گراست) داشت و این عدم سازگاری به موقعیت ATM لطمهٔ بسیار فراوانی زد.

در آخر آنکه اترنت ظرفیت پیشرفت و بهبود فراوانی از خود نشان داد. سرعت آن چند ده برابر شد، هاب و سوئیچ عرضه گردید ولی با تمام این تغییرات نیازی به تغییر در نرم‌افزار نبود. تصور کنید وقتی یک عرضهٔ کنندهٔ محصولات شبکه، تجهیزات مفصلی را ارائه کرده و می‌گوید: «این شبکهٔ جدید و عالی را برای شما تدارک دیده‌ایم. برای استفاده از آن باید زحمت بکشید و سخت‌افزارهای قبلی خود را دور ریخته و تمام نرم‌افزارهای خود را از نو بنویسید!! او در فروش شبکهٔ خود قطعاً به مشکل برخواهد خورد!!! اترنت چنین مشکلی نداشت. شبکه‌هائی مثل FDDI، Fibre Channel و ATM در زمان عرضه بسیار سریعتر از اترنت بودند ولیکن هیچکدام از آنها با اترنت سازگار نبودند، بسیار پیچیده‌تر و مدیریت آنها دشوارتر از اترنت بود. سرانجام وقتی سرعت اترنت بهبود یافت این شبکه‌ها دیگر هیچ مزیتی نداشتند و سریعاً از میان رفتند؛ البته بجز ATM که آن هم در هستهٔ سیستمهای تلفنی به کارگرفته شده بود.

٤-٤ شبکه‌های محلی بی‌سیم

اگر چه اترنت در سطح گسترده‌ای رایج است ولی رقیب جدیدی برای آن در حال ظهور است. شبکه‌های بی‌سیم

در حال رواج هستند و بطور فزاینده‌ای در دفاتر اداری، فرودگاه‌ها و دیگر مکانهای عمومی به کار گرفته می‌شوند. شبکه‌های بی‌سیم به نحوی که در شکل ۱-۳۵ دیدیم به دو روش پیکربندی می‌شوند: «در کنار یک ایستگاه ثابت» و «بدون ایستگاه ثابت». استاندارد 802.11 هر دوی این پیکربندیها را مدّ نظر قرار داده و برای هر دو ، تدارک لازم را دیده است.

در بخش ۱-۵-۴ پیش‌زمینه‌ای از 802.11 ارائه نمودیم. حال زمان آن رسیده تا نگاهی دقیقتر به این تکنولوژی بیندازیم. در بخشهای آتی نگاهی به پشته پروتکلی ، تکنیکهای ارسال رادیوئی در لایهٔ فیزیکی ، پروتکل زیرلایهٔ MAC ، ساختار فریم و خدمات ارائه شده در این شبکه خواهیم انداخت. برای کسب آگاهی بیشتر در خصوص 802.11 به مراجع زیر مراجعه نمایید:

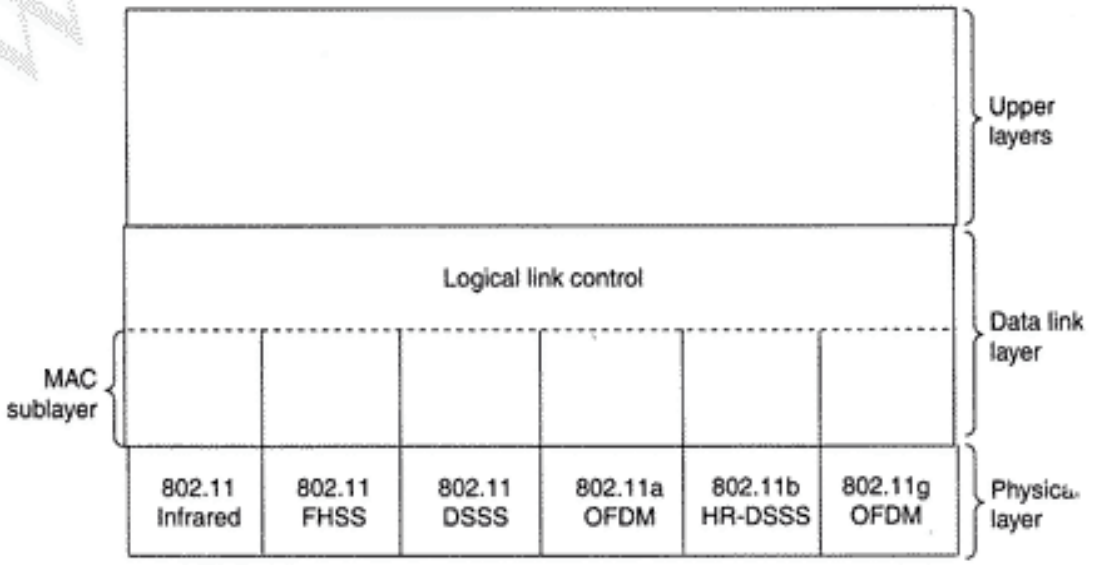
(Crow et al., 1997; Geier, 2002; Heegard et al., 2001; Kapp, 2002; O'Hara & Petric, 1999; and Severance, 1999)

برای شنیدن حقیقت درست و مسلّم این شبکه، باید مستقیماً به استاندارد IEEE 802.11 مراجعه نمایید.

۴-۴-۱ پشتهٔ پروتکلی 802.11

پروتکلهایی که در استانداردهای سری ۸۰۲ و از جمله اترنت به کار گرفته شده مشترکات ساختاری فراوان دارند. شمای کلّی پشتهٔ پروتکلی 802.11 ، در شکل ۴-۲۵ نمایش داده شده است. لایهٔ فیزیکی به خوبی هم تراز با لایهٔ فیزیکی از مدل OSI است در حالیکه لایهٔ پیوند داده از دو لایهٔ مستقل تشکیل شده است. در 802.11 زیرلایهٔ MAC (زیرلایهٔ کنترل دسترسی به کانال) ، چگونگی دسترسی و تخصیص کانال ، یعنی ایستگاهی را که باید در ادامه ارسال داشته باشد، مشخص می‌کند. برروی آن، زیرلایهٔ LLC قرار می‌گیرد که وظیفه اصلی آن مخفی کردن تفاوتهای موجود درگونه‌های مختلف ۸۰۲است، بگونه‌ای که این تفاوتها برای لایهٔ شبکه [لایهٔ سوم در مدل OSI] مخفی و غیر قابل‌تشخیص باشد. در همین فصل وقتی اترنت را بررسی می‌کردیم، LLC را نیز مطالعه نمودیم و نیازی به تکرار آنها نیست.

در سال ۱۹۹۷ استاندارد 802.11 سه تکنیک مختلف انتقال رادیویی، برای به کارگیری در لایهٔ فیزیکی معرفی کرد. مثلا روش مبتنی بر امواج مادون قرمز، بسیار شبیه به روشی است که در کنترل از راه دور تلویزیونها به کاررفته



شکل ۴-۲۵. بخشی از پشتهٔ پروتکلی 802.11.

است. در روش دیگر از امواج رادیوئی برد کوتاه و از تکنیک‌هائی به نام FHSS و DSSS بهره گرفته شده است. هر دوی این روشها از محدوده‌ای در طیف فرکانس استفاده می‌کنند که نیازی به اخذ مجوز از دولت ندارد (باند 2.4 GHZ). به عنوان مثال دربازکن‌های کنترل از راه دور نیز از همین باند فرکانسی استفاده می‌کنند لذا کامپیوتر کیفی شما ممکن است در حین استفاده از کانال، خودش را رقیب درب گاراژ شما ببیند!!! تلفن‌های بی‌سیم و اجاق‌های مایکروویو نیز از همین باند فرکانسی بهره گرفته‌اند.

تکنیک‌های ارسال رادیوئی با نرخ ۱ تا ۲ مگابیت در ثانیه و با توان بسیار کمی عمل می‌کنند تا تداخل رادیوئی این ابزارها با یکدیگر حداقل باشد. در سال ۱۹۹۹ دو تکنیک جدید معرفی شد تا پهنای باند (نرخ ارسال) آن افزایش یابد. این دو تکنیک جدید OFDM و HR-DSSS نامیده شده‌اند و به ترتیب با سرعتهای ۵۴ و ۱۱ مگابیت بر ثانیه عمل می‌کنند. در سال ۲۰۰۱ گونهٔ دوّمی از مدولاسیون OFDM ولی در باند فرکانسی متفاوت نسبت به OFDM اولیه، معرفی شد. در ادامه بطور مختصر آنها را بررسی خواهیم کرد. از دیدگاه فنی این تکنیکها به لایهٔ فیزیکی تعلق دارند و باید در فصل دوّم بررسی می‌شدند ولی از آنجایی که این تکنیکها به LAN و خصوصاً زیرلایهٔ 802.11 MAC وابسته هستند ، در این بخش بدانها پرداخته‌ایم.

۲.٤.٤ لایهٔ فیزیکی در 802.11

تمام پنج تکنیک انتقال رادیویی، این امکان را فراهم کرده‌اند که یک فریم MAC از ایستگاهی به ایستگاه دیگر منتقل شود. تفاوتهای آنها در تکنولوژی به کاررفته و سرعت قابل حصول آنهاست. پرداختن به جزئیات این روشها از حوصلهٔ این کتاب خارج است ولیکن چند کلمه صحبت در مورد آنها و بخصوص معرفی کلمات کلیدی آن می‌تواند به علاقمندان کمک کند تا بتوانند برای کسب آگاهی بیشتر، در اینترنت یا مراجع دیگر جستجو کنند.

در گزینهٔ «امواج مادون قرمز» از امواج پخشی (Diffused) با طول موج ۰/۸۵ یا ۰/۹۵ میکرون بهره گرفته شده است. در این روش سرعتهای ۱ و ۲ مگابیت در ثانیه مجاز می‌باشد. در نرخ 1 Mbps از روش کدینگ خاصی به نام Gray Code استفاده شده که در آن گروه‌های ۴ بیتی به یک کلمهٔ کد ۱۶ بیتی تبدیل می‌شوند، به نحوی که در این کلمهٔ ۱۶ بیتی تنها یک بیت ۱ و پانزده بیت ۰ وجود دارد. این کد دارای این ویژگی اساسی است که خطائی کوچک در سنکرونیزاسیون زمان، تنها یک بیت خطا در خروجی ایجاد خواهد کرد. در سرعت 2 Mbps ، دو بیت اخذ و یک کد چهار بیتی تولید می‌شود که در آن تنها یک بیت ۱ وجود دارد. (یعنی هر دو بیت به یکی از چهار حالت 0001 , 0010 , 0100 , 1000 نگاشته می‌شود.) سیگنال‌های مادون قرمز نمی‌توانند در موانعی مثل دیوار نفوذ کنند لذا سلولهائی که در اتاقهای مختلف ایجاد می‌شوند کاملاً از هم جدا و تفکیک‌شده هستند. علیرغم این، بدلیل نرخ ارسال پائین (و این حقیقت که نور خورشید، امواج مادون قرمز را در خود غرق و محو می‌کند) از این گزینه استقبال چندانی نشد.

در FHSS (Frequency Hopping Spread Spectrum) از ۷۹ کانال مستقل استفاده شده که هر یک از این کانالها 1 MHz پهنای باند دارند و از پائینترین فرکانس باند 2.4 GHz ISM شروع می‌شوند. برای مشخص کردن دنبالهٔ فرکانسهائی که باید بدانها پرش شود از یک مولد اعداد شبه‌تصادفی استفاده شده است. مادامیکه تمام ایستگاه‌ها در الگوریتم مولد اعداد از نقطهٔ شروع (Seed) یکسانی استفاده کنند و از لحاظ زمانی با هم سنکرون باشند همگی بطور همزمان به فرکانسهای یکسانی پرش خواهند کرد. مدت زمانی که ایستگاه‌ها در یک فرکانس خاصّ باقی می‌مانند، اصطلاحاً dwell time نامیده می‌شود و پارامتری قابل تنظیم است ولیکن باید کمتر از ۴۰۰ میلی‌ثانیه باشد. استفادهٔ تصادفی از باندهای فرکانسی، روش مناسبی برای تخصیص کانال بروشی غیرمعمول در باند ISM است. این ویژگی کم و بیش به امنیت اطلاعات کمک خواهد کرد چرا که اگر یک اخلالگر ترتیب پرشهای فرکانس یا پارامتر Dwell time را نداند، نمی‌تواند اطلاعات کانال را استراق سمع کند. در فواصل دور،

محوشدگی سیگنال بدلیل «چندمسیره شدن سیگنال» (Multipath Fading) اشکال عمده‌ای به حساب می‌آید و خوشبختانه FHSS در مقابله با این مشکل، موفق عمل می‌کند. همچنین این روش نسبت به تداخل رادیوئی، نسبتاً حساس نیست و برای ایجاد لینک بین ساختمانها بسیار مناسب خواهد بود. بزرگترین اشکال این روش پهنای باند کم آنست.(1 Mbps)

سومین روش مدولاسیون یعنی DSSS (Direct Sequence Spread Spectrum) نیز به یکی از نرخ‌های ۱ یا ۲ مگابیت برثانیه محدود شده است. تکنیک به کار رفته در DSSS تا حدودی مشابه به سیستم CDMA است که در بخش ۲.۶.۲ بررسی شد ولیکن از برخی جهات با آن تفاوتهایی دارد. هر بیت در قالب یازده Chips ارسال می‌شود که به دنبالهٔ بارکر (Barker Sequence) مشهور است. در این روش از مدولاسیون تغییر فاز (Phase Shift) با نرخ تغییر 1 Mbaud استفاده شده که وقتی در نرخ 1 Mbps عمل می‌کند در هر تغییر فاز یک بیت ولی وقتی در نرخ 2 Mbps عمل می‌کند در هر تغییر فاز، دو بیت منتقل می‌گردد. برای سالهای متمادی سازمان FCC [سازمان تخصیص فرکانس] فقط اجازه می‌داد که تجهیزات مخابرات بی‌سیم در ایالات متحده، صرفاً از باند ISM و طیف گسترده (Spread Spectrum) استفاده کنند ولی ظهور تکنولوژیهای جدید، به لغو این قانون در سال ۲۰۰۲ انجامید.

اوّلین شبکهٔ محلی بی‌سیم پرسرعت یعنی 802.11a ، با بهره‌گیری از مدولاسیون OFDM[1] در باند فرکانسی 5-GHz عمل می‌کرد تا به سرعت 54 Mbps دست یابد. اگر بخواهیم با استعارات FDM سخن بگوئیم، در OFDM از ۵۲ زیرکانال فرکانسی استفاده شده که ۴۸ تا از آنها برای داده و ۴ تا برای سنکرونیزاسیون کاربرد دارد و بی‌شباهت به ADSL نیست. از آنجایی که ارسال، بطور همزمان برروی فرکانسهای متفاوتی انجام می‌شود لذا این روش گونه‌ای از روشهای مبتنی بر «طیف گسترده» محسوب می‌شود ولیکن با روشهای CDMA یا FHSS کاملاً متفاوت است. تقسیم سیگنال به تعداد بسیار زیادی باند باریک در مقایسه با استفاده از یک باند عریض و واحد، مزایای بسیار مهمی در بردارد که از جمله می‌توان به ایمنی بیشتر در مقابل تداخل امواج باند باریک و امکان استفاده از باندهای غیر مجاور (noncontiguous band) اشاره کرد. در این روش از سیستم کدینگ پیچیده‌ای مبتنی بر مدولاسیون تغییر فاز برای سرعت زیر 18 Mbps و مدولاسیون QAM برای سرعت‌های بالاتر استفاده شده است. در سرعت 54 Mbps ، ۲۱۶ بیت داده به سمبول‌های ۲۸۸ بیتی کد می‌شود. بخشی از انگیزه‌های خلق OFDM سازگاری آن با سیستم اروپائی HiperLAN/2 بوده است. (Doufexi et al., 2002) این روش دارای کارآئی بسیار بالائی در استفاده از طیف فرکانسی (برحسب bits/HZ) بوده و ایمنی خوبی در مقابل پدیدهٔ «محوشدگی ناشی از مسیرهای چندگانه» دارد.

نهایتاً به HR-DSSS (High Rate Direct Spread Spectrum) می‌رسیم که روشی دیگر مبتنی بر تکنیک طیف گسترده است و با به کارگیری 11 million chips/sec در باند 2.4 GHz ، به نرخ ارسال یازده مگابیت در ثانیه رسیده است. این روش 802.11b نامیده شده ولیکن دنباله روی 802.11a نبوده است. در این روش از نرخهای ۱، ۲، ۵/۵ و ۱۱ مگابیت در ثانیه حمایت می‌شود. دو نرخ ارسال پائین [یعنی ۱ و ۲ مگابیت بر ثانیه] به ترتیب از سیگنالی با نرخ تغییر 1 Mbaud/sec بهره می‌گیرد (یعنی در هر تغییر فاز، ۱ یا ۲ بیت کد می‌شود). در HR-DSSS به منظور سازگاری با DSSS ، از روش مدولاسیون تغییر فاز بهره گرفته شده است. برای نرخ ارسال سریعتر (۵/۵ و ۱۱ مگابیت بر ثانیه) از سیگنالی با نرخ تغییر 1.375 Mbaud/sec استفاده شده و در هر تغییر به ترتیب ۴ یا ۸ بیت کد می‌شود؛ کدها از نوع Walsh/Hadamard هستند. نرخ ارسال به صورت پویا و در خلال عملیات شبکه تعیین

---

۱. Orthogonal Frequency Division Multiplexing

می‌شود تا سرعت بهینه بر اساس شرایط فعلی حاکم بر شبکه (شامل نویز محیط و بار) تنظیم گردد. در عمل، سرعت شبکه 802.11b تقریباً همیشه 11 Mbps است. اگر چه سرعت 802.11b از سرعت 802.11a کمتر می‌باشد ولیکن بُرد این شبکه حدوداً هفت برابر بیشتر است که این ویژگی در بسیاری از محیطها اهمیت بسزائی دارد.

نسخهٔ بهبود یافتهٔ 802.11b یعنی 802.11g ، در نوامبر سال ۲۰۰۱ (پس از کشمکش فراوان بر سر آنکه از کدام تکنولوژی استفاده شود) به تائید IEEE رسید. این استاندارد از مدولاسیون OFDM (به کار رفته در 802.11a) بهره می‌گیرد ولیکن مثل 802.11b در باند 2.4 GHz عمل می‌کند. از نظر تئوری این سیستم می‌تواند در سرعت 54 Mbps عمل کند، ولیکن هنوز روشن نیست که آیا این سرعت در عمل نیز محقق خواهد شد یا خیر. بدین ترتیب کمیتهٔ 802.11 سه شبکهٔ محلی پرسرعت بی‌سیم معرفی کرده است. 802.11a ، 802.11b و 802.11g (به سه شبکهٔ کندتر فعلاً کاری نداریم) شاید این سوال درست به ذهن‌شما خطور کند که آیا تعریف سه استاندارد متفاوت کار درستی است؟ شاید عدد ۳، عدد طلائی شانس باشد!

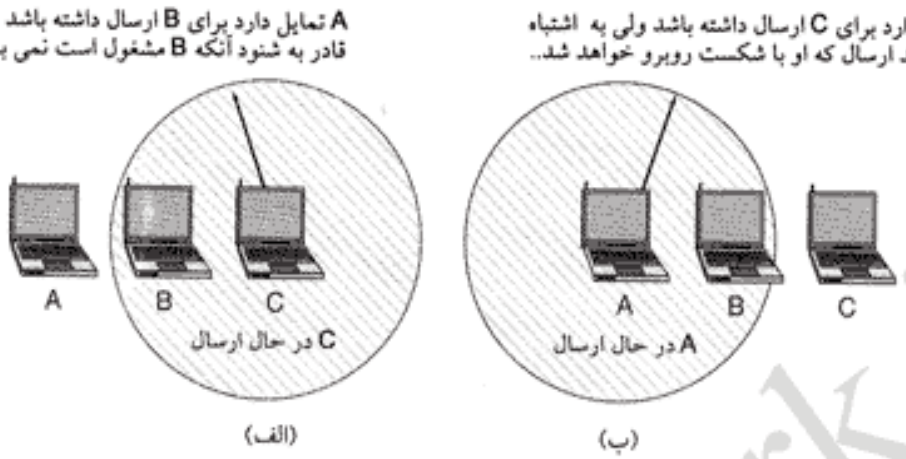
٤-٤-٣ پروتکل زیرلایهٔ MAC در 802.11

حال اجازه بدهید از فضای مهندسی برق به سرزمین مهندسی کامپیوتر برگردیم. پروتکل زیرلایهٔ MAC در 802.11 کاملاً با اترنت تفاوت دارد زیرا شرایط حاکم بر محیطهای بی‌سیم در مقایسه با سیستمهای سیم‌دار، دارای پیچیدگیهای ذاتی است. در اترنت یک ایستگاه منتظر می‌ماند تا کانال آزاد شود؛ سپس ارسال خود را شروع می‌کند. اگر در خلال ارسال ۶۴ بایت اوّل فریم، هیچ نویز شدیدی برنگشت، می‌توان اعتقاد داشت که فریم به درستی تحویل مقصد شده است. در شبکهٔ بی‌سیم چنین وضعیتی حاکم نیست.

برای شروع باز هم یادآور می‌شویم که مشکل ایستگاه مخفی (که قبلاً تشریح و مجدداً در شکل ۴-۲۶ به تصویر کشیده شده است) ایجاد اشکال خواهد کرد. از آنجایی که تمام ایستگاه‌ها در برد رادیوئی یکدیگر نیستند فلذا ارسال سیگنال در بخشی از یک سلول، ممکن است در ناحیهٔ دیگری از همان سلول قابل دریافت و شنود نباشد. در این مثال ایستگاه C در حال ارسال به ایستگاه B است ولی اگر A کانال را بررسی و شنود کند هیچ چیزی نمی‌شنود و به غلط نتیجه می‌گیرد که باید ارسال برای B را آغاز کند. [شکل ۴-۲۶-الف]

همچنین عکس این مشکل نیز وجود دارد. در شکل ۴-۲۶-ب ایستگاه B می‌خواهد فریمی را برای C بفرستد و به همین دلیل به کانال گوش می‌دهد. وقتی سیگنال در حال انتقال را شنود می‌کند به غلط نتیجه می‌گیرد که نباید برای ایستگاه C چیزی بفرستد ولی علیرغم آنکه A در حال ارسال برای D است (D در شکل نشان داده نشده) ایستگاه B می‌تواند برای C ارسال داشته باشد. مضاف براین، ارتباط بی‌سیم اغلب ماهیت «دوطرفهٔ غیرهمزمان» (Half Duplex) دارد بدین معنا که ایستگاه‌ها نمی‌توانند در حین ارسال و بطور همزمان، کانال را برای آگاهی از وضعیت تصادم برروی همان باند فرکانسی شنود کنند. [این کار در کانالهای سیمی به راحتی امکان‌پذیر است] در نتیجه 802.11 ، برخلاف اترنت از CSMA/CD استفاده نمی‌کند.

برای رفع این مشکلات، استاندارد 802.11 از دو روش عملکرد پشتیبانی می‌کند: در اولین روش که DCF نامیده می‌شود (Distributed Coordination Function) هیچ‌گونه کنترلی مرکزی وجود ندارد (و از این دیدگاه مشابه با اترنت است). در روش دیگر که PCF (Point Coordination Function) نامیده می‌شود، برای کنترل و نظارت بر کلیهٔ فعالیتهای درون هر سلول ، از یک ایستگاه ثابت استفاده شده است. در پیاده‌سازی استاندارد 802.11، باید از DCF پشتیبانی شود در حالیکه حمایت از PCF اختیاری است. به ترتیب این دو روش را تشریح خواهیم کرد.



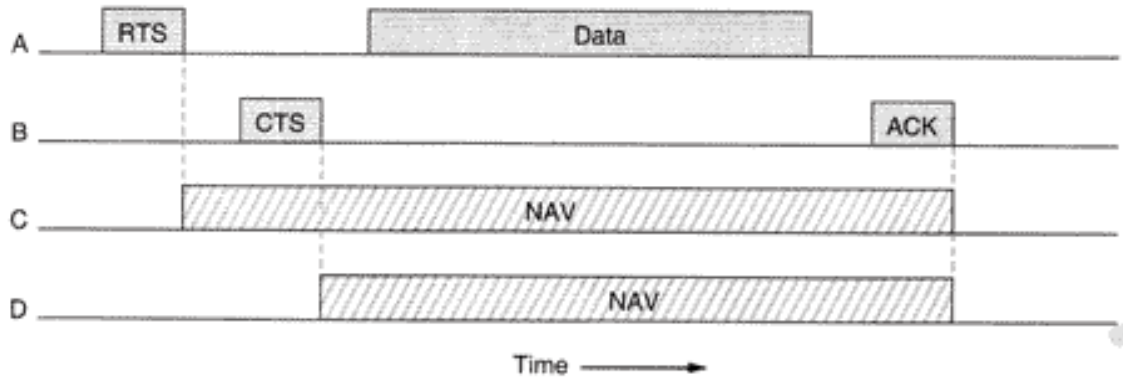
شکل ۴-۲۶. (الف) مشکل ایستگاه مخفی (ب) مشکل ایستگاه آشکار.

وقتی از حالت DCF استفاده می‌شود، 802.11 از پروتکلی به نام CSMA/CA[1] بهره می‌گیرد. در این پروتکل هم کانال فیزیکی و هم کانال مجازی شنود می‌شوند. CSMA/CA از دو عملکرد متفاوت پشتیبانی می‌کند: در روش اوّل وقتی یک ایستگاه می‌خواهد فریمی را ارسال کند ، ابتدا به شنود می‌پردازد و اگر کانال آزاد بود ارسال خود را آغاز می‌کند. در حین ارسال فریم، **کانال شنود نمی‌شود** و کلّ فریم منتقل می‌گردد، در حالیکه ممکن است این فریم، بدلیل تداخل رادیوئی در گیرنده از بین برود. برعکس، اگر کانال اشغال باشد فرستنده ارسال خود را تا زمان آزاد شدن کانال به تعویق انداخته و سپس شروع می‌کند. در صورت بروز تصادم ، ایستگاه‌های تصادم‌کننده به اندازهٔ یک زمان تصادفی (که بر اساس الگوریتم عقب‌گرد نمایی تعیین می‌شود) منتظر مانده و از نو تلاش می‌کنند. [تا اینجا همه چیز شبیه به پروتکل CSMA/CD است.]

روش دیگر به کاررفته در CSMA/CA مبتنی بر MACAW است که از روش «شنود کانال مجازی» بهره می‌گیرد. در مثال شکل ۴-۲۷ ، ایستگاه A می‌خواهد فریمی را برای B بفرستد. C ، ایستگاهی در برد ایستگاه A است (شاید C هم در برد ایستگاه B باشد ولی مهم نیست). D ، ایستگاهی در برد B است ولی در برد A قرار ندارد. پروتکل زمانی آغاز می‌شود که A تصمیم به ارسال داده برای B می‌گیرد. او کارش را با ارسال یک فریم کوتاه RTS برای B آغاز و تقاضای مجوز ارسال فریم می‌نماید. هرگاه B این تقاضا را دریافت کند (احتمالاً) تصمیم به صدور مجوز می‌گیرد؛ در این حالت فریم CTS را برمی‌گرداند. پس از دریافت فریم CTS ، ایستگاه A ارسال فریم خود را شروع کرده و یک زمان‌سنج خاصّ به نام ACK-Timer را روشن می‌کند. پس از دریافت فریم داده، ایستگاه B با ارسال فریم ACK به مبادله داده خاتمه می‌دهد. اگر زمان‌سنج ، قبل از آنکه فریم ACK باز گردد، منقضی شود [یعنی فریم ACK در زمان معقولی برنگردد] کلّ این فرآیند باید از نو اجرا شود.

حال بیائید از دیدگاه ایستگاه‌های C و D به این فرآیند مبادلهٔ فریم، نگاه کنیم. C ایستگاهی است که در برد A است و طبعاً RTS را دریافت می‌کند. اگر اینگونه باشد متوجه خواهد شد که شخص دیگری بزودی ارسال خود را آغاز خواهد کرد، فلذا برای احتیاط کامل، تا تکمیل عملیات مبادلهٔ داده، از ارسال هر چیزی اجتناب می‌کند. C می‌تواند از طریق اطلاعاتی که از فریم RTS بدست می‌آید، کلّ زمان ارسال را تخمین بزند (با احتساب زمان برگشت فریم ACK) ، لذا برای خودش فرض می‌کند که در خلال زمانی که در شکل ۴-۲۷ با عنوان NAV

۱.CSMA with Collision Avoidance



شکل ۴ـ۲۷. کاربرد کانال مجازی در روش CSMA/CA.

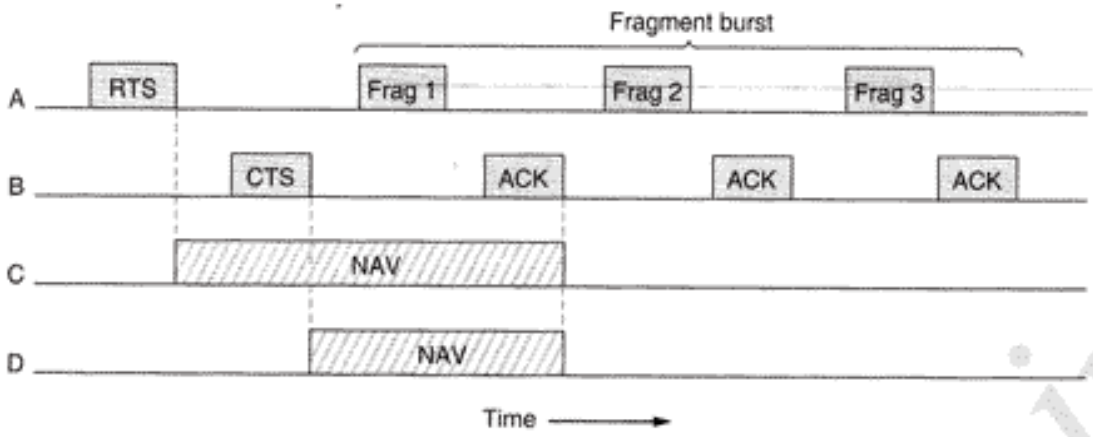
(Network Allocation Vector) مشخص شده، کانال مجازی مشغول است. ایستگاه D فریم RTS را نمی‌شنود ولیکن CTS را خواهد شنید لذا او هم برای خودش به اندازه زمان NAV کانال را مشغول فرض می‌کند. دقت کنید که سیگنال NAV به هیچ وجه ارسال نمی‌شود بلکه فقط یک یادداشت داخلی است که ایستگاه را در مدت زمان معینی ساکت نگه می‌دارد.

برخلاف شبکه‌های سیمی، شبکه‌های بی‌سیم غیرقابل‌اعتماد و نویزی هستند چراکه دستگاه‌های مختلف مثل اجاقهای مایکروویو (که آنها نیز در باند ISM کار می‌کنند) نویز قوی و مخرب تولید می‌نمایند. در نتیجه، احتمال انتقال موفق فریم با افزایش طول فریم کاهش خواهد یافت. هرگاه احتمال خرابی یک بیت p باشد احتمال آنکه یک فریم n بیتی، کامل و سالم دریافت شود $(1-p)^n$ خواهد بود. برای مثال اگر $p=10^{-4}$ باشد احتمال سالم رسیدن یک فریم کامل اترنت (۱۲۱۴۴ بیتی) کمتر از ۳۰ درصد است. برای $p=10^{-5}$، از هر ۹ فریم یکی خراب خواهد شد. برای $p=10^{-6}$ حدود یک درصد از کل فریمها آسیب خواهد دید که تقریباً معادل یک دوجین فریم در ثانیه خواهد بود. بطور خلاصه اگر یک فریم بزرگ باشد شانس کمتری در سالم رسیدن آن به مقصد وجود دارد و احتمالاً باید مجدداً ارسال شود.

برای کاهش مشکل کانالهای نویزی، 802.11 اجازه داده که هر فریم به قطعات کوچکتری تقسیم شده و هر کدام کد کشف خطای خود را داشته باشند. قطعات بطور مجزا شماره‌گذاری شده و دریافت آن به روش «**توقف و انتظار**» (Stop & Wait) تائید می‌شود. (به عبارت دیگر فرستنده قطعه شمارهٔ k+1 را نخواهد فرستاد مگر آنکه پیام ACK قطعهٔ k‌ام –مبنی بر دریافت صحیح آن– دریافت شود.) پس از آنکه به کمک RTS و CTS، کانال در اختیار ایستگاهی قرار گرفت، طبق شکل ۴–۲۸، آن ایستگاه می‌تواند متوالیاً چندین قطعهٔ مستقل، ارسال کند. دنبالهٔ قطعات متوالی، اصطلاحاً Fragment Burst نامیده می‌شود.

عملیات قطعه‌قطعه‌سازی فریمها، کارائی مفید شبکه را افزایش خواهد داد چرا که به جای ارسال مجدد کل فریم فقط قطعات کوچکی که در اثر خطای کانال خراب شده‌اند از نو ارسال خواهند شد. طول هر قطعه به صورت قطعی و ثابت در استاندارد تعیین نشده است بلکه جزو پارامترهای قابل تنظیم هر سلول محسوب و توسط ایستگاه ثابت (Base Station) تنظیم می‌شود. مکانیزم NAV ایستگاه‌ها را آنقدر ساکت نگاه می‌دارد تا زمانیکه دریافت کلّ فریم تایید شود، ولیکن برای آنکه ایستگاه‌ها آنقدر تامل کنند تا دنبالهٔ کل قطعات فریم (Fragment Burst) بدون تداخل ارسال شود، مکانیزم دیگری که در زیر تشریح شده، بکار می‌رود.

تمام توضیحات فوق در حالت DCF از استاندارد 802.11 قابل اعمال و صائب است. یادآوری می‌کنیم که در حالت DCF هیچ کنترل و نظارت مرکزی وجود ندارد و تمام ایستگاه‌ها مشابه با آنچه که در اترنت اتفاق می‌افتد برای بدست آوردن کانال رادیوئی رقابت می‌کنند. در حالت دیگر یعنی PCF، یک ایستگاه ثابت یکی‌یکی به



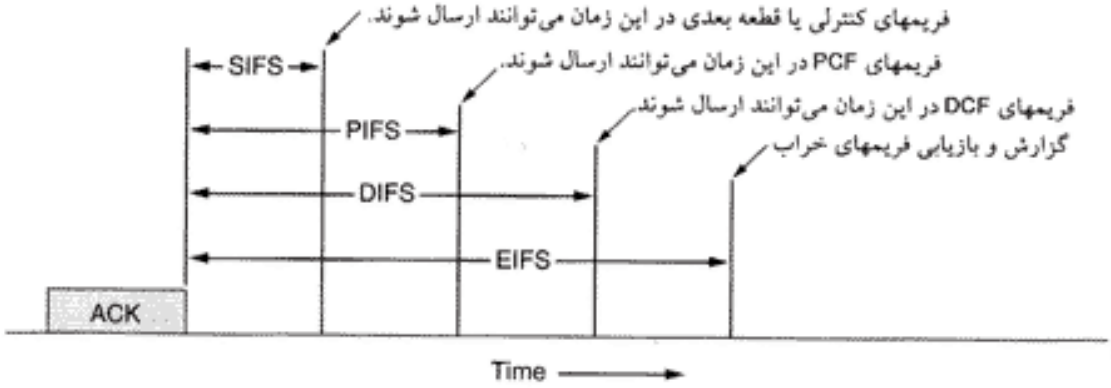
شکل ۴-۲۸. ارسال انفجاری چند قطعه.

ایستگاه‌ها سرکشی کرده و از آنها سوال می‌کند که آیا فریمی جهت ارسال دارند یا خیر. از آنجایی که در حالت PCF بر تمام تقاضاهای ارسال فریم به صورت مرکزی نظارت می‌شود لذا هیچگونه تصادم اتفاق نخواهد افتاد. استاندارد 802.11 ، مکانیزم سرکشی به ایستگاه‌ها را تعیین کرده است ولیکن دفعات سرکشی ، ترتیب سرکشی یا حتی تصمیم‌گیری در خصوص آنکه آیا همهٔ ایستگاه‌ها باید به یک اندازه از شبکه سرویس بگیرند، در استاندارد تعریف و تعیین نشده است.

مکانیزم اصلی در سرکشی به ایستگاه بدین نحو است که یک ایستگاه فریم خاصی به نام Beacon Frame (فریم فانوس دریائی) را بطور متناوب در فضای پیرامون خود منتشر می‌کند. (ده تا صد بار در هر ثانیه) «فریم فانوس دریائی» شامل پارامترهای مختلف سیستم مثل: «ترتیب پرش فرکانسی» (Hopping Sequence) و پارامتر dwell time (برای مدولاسیون FHSS) و پارامتر سنکرون‌سازی سیگنال ساعت و نظایر آن، می‌باشد. همچنین توسط این فریم از ایستگاه‌های جدید دعوت می‌شود تا به منظور سرکشی شدن ثبت‌نام کنند. پس از آنکه ایستگاهی برای دریافت خدمات سرکشی با نرخ معین، ثبت‌نام کرد، این تضمین را دارد که بخش معینی از پهنای باند شبکه به او اختصاص داده خواهد شد و بدین ترتیب آن ایستگاه می‌تواند **«کیفیت خدمات»** (Quality of Service) خود را تضمین کند. [برای مطالعه در خصوص کیفیت خدمات به بخش ۱-۳-۳ و فصل ۵ مراجعه کنید.]

طول عمر باطری یکی از مسائل مهم در ابزارهای همراه و بی‌سیم بوده و هست؛ به همین دلیل در استاندارد 802.11 به مسئله مدیریت توان مصرفی، توجه ویژه‌ای شده است. خصوصاً در حالت PCF ایستگاه ثابت می‌تواند ایستگاه همراه را به «حالت استراحت» (Sleep State) ببرد؛ تا زمانیکه بطور مشخص ایستگاه ثابت یا کاربری دیگر آن را از این حالت بیرون آورده و بتواند فعالیت عادّی خود را از سربگیرد. مجبور کردن یک ایستگاه به استراحت، بدین معناست که ایستگاه ثابت مسئولیت دارد تمام فریمهائی را که برای ایستگاه غیرفعال (در حال استراحت) ارسال می‌شود، دریافت و بافر کند. بعداً این فریمها به صورت یکجا تحویل خواهد شد.

در درون یک سلول می‌توان بطور همزمان هم حالت PCF و هم حالت DCF را به کار گرفت. در نگاه اوّل ممکن است به نظر برسد که نظارت مرکزی و نظارت توزیع‌شده بطور همزمان میسّر نباشد ولی در استاندارد 802.11 برای رسیدن به چنین هدفی راهکاری مناسب اندیشیده شده است. برای این کار، بازه‌های زمانی بین فریمها بدقت تعریف می‌شود. پس از آنکه یک فریم ارسال شد و قبل از آنکه ایستگاهی بتواند فریم بعدی را ارسال نماید به مدّتی **«زمان مرده»** نیاز است. در این زمان مرده، چهار بازهٔ زمانی مجزا با اهداف خاص، تعریف



شکل ۴ـ۲۹. فاصلهٔ زمانی بین فریمها در 802.11

شده است. این چهار بازهٔ زمانی در شکل ۴-۲۹ نشان داده شده است.

کوتاه‌ترین بازهٔ زمانی، بازهٔ SIFS (Short InterFrame Spacing) است. این بازهٔ زمانی به ایستگاه‌ها فرصت می‌دهد تا به ارسال فریمهای کنترلی خاص بپردازند. در این زمان ایستگاه‌ها اجازه می‌یابند عملیاتی مثل ارسال CTS (در پاسخ به RTS)، ارسال فریم ACK در پاسخ به یک فریم کامل یا یک قطعه از فریم، ارسال یک قطعه از دنبالهٔ قطعات (بدون ارسال RTS مجدّد) یا نظائر این را انجام بدهند.

همیشه فقط یک ایستگاه است که پس از زمان SIFS، به منظور پاسخ‌دهی [و ارسال فریم کنترلی مناسب] حقّ ارسال دارد. اگر ایستگاه مربوط نتواند از این فرصت استفاده کند و زمان PIFS (PCF InterFrame Spacing) منقضی شود، ایستگاه ثابت می‌تواند «فریم فانوس دریائی» (Beacon) یا «فریم سرکشی» ارسال نماید. این مکانیزم اجازه می‌دهد که ایستگاه در حال انتقال فریم یا دنبالهٔ قطعات، بدون آنکه ایستگاه دیگری در این میان مداخله کند ارسال فریم خود را به پایان برساند در حالیکه ایستگاه ثابت نیز این فرصت را خواهد داشت که وقتی ایستگاه قبلی کارش را به اتمام رساند کانال را بدون رقابت با کاربران متمایل به ارسال تصرف نماید.

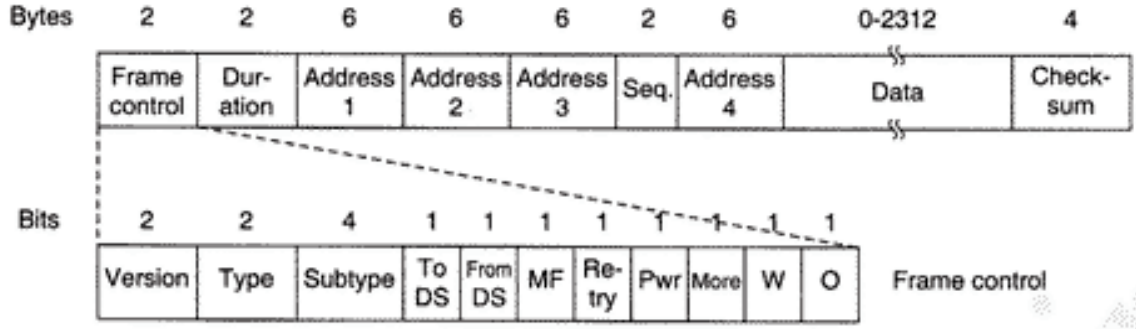
هرگاه ایستگاه ثابت چیزی برای ارسال نداشته باشد و زمان DIFS (DCF InterFrame Spacing) منقضی شود هر ایستگاه می‌تواند بخت خود را در تصرف کانال و ارسال فریم بیازماید. در این لحظه، برای در اختیار گرفتن کانال، روش معمولی رقابت و در صورت تصادم الگوریتم عقب‌گرد نمایی اعمال می‌شود.

آخرین بازهٔ زمانی یعنی EIFS (Extended InterFrame Spacing) مورد استفادهٔ ایستگاهی قرار می‌گیرد که یک فریم خراب یا فریمی ناشناس دریافت کند تا بتواند این مسئله را گزارش بدهد. دلیل اصلی آنکه به این بازهٔ زمانی کمترین اولویت داده شده [آخرین بازهٔ زمانی است] آن بوده که چون گیرندهٔ نمی‌داند چه اتفاقی در جریان است لذا باید مدت زمان قابل توجهی صبر کند تا از هرگونه مداخله در گفتگوی دو ایستگاه اجتناب نماید.

٤ـ٤ـ٤ ساختار فریم 802.11

استاندارد 802.11 سه ردهٔ مختلف فریم برای ارسال برروی کانال تعریف کرده است: «**فریم داده**»، «**فریم کنترلی**» و «**فریم مدیریتی**». هر یک از این فریمها سرآیند (Header) خاصّ خود را دارند که در هر سرآیند، فیلدهائی جهت استفاده در زیرلایهٔ MAC تعریف شده است. مضاف بر این سرآیندهائی جهت به کارگیری در لایهٔ فیزیکی تعریف گردیده که اغلب تکنیک‌های مدولاسیون [و پارامترهای مخابراتی] را مشخص می‌کنند، لذا در اینجا بدانها نخواهیم پرداخت.

قالب فریم داده، در شکل ۴-۳۰ نشان داده شده است. در ابتدا «فیلد کنترل» ظاهر شده که این فیلد خودش دارای یازده فیلد فرعی است. اولین زیرفیلد، شمارهٔ نسخهٔ پروتکل (Protocol Version) را مشخص می‌کند؛



شکل ۴ـ۳۰. فریمهای داده در 802.11.

بدین ترتیب در آن واحد و در یک سلول مشابه، به کارگیری دو پروتکل متفاوت ممکن خواهد بود. در ادامه، زیرفیلد دو بیتی Type آمده که نوع فریم را [اعم از فریم داده، کنترلی و مدیریتی] مشخص می‌نماید؛ پس از آن زیرفیلد Subtype قرار گرفته که مشخصات دقیقتر فریم (مثل RTS و CTS) را تعریف می‌کند. سپس بیتهای ToDS و FromDS آمده که مشخص می‌کنند که آیا فریم از یک «سیستم توزیع درون‌سلولی» (مثلاً شبکه اترنت) بیرون آمده یا بدانجا رهسپار است. بیت MF نشان می‌دهد که هنوز قطعاتی از فریم در پیش رو هستند [و هنوز قطعاتی از فریم باقیست.] بیت Retry نشانگر آنست که فریم جاری قبلاً یکبار ارسال شده است. ایستگاه ثابت بکمک بیت Power Management (که در شکل با نماد Pwr نشان داده شده است)، ایستگاه متحرک را به حالت استراحت (Sleep) برده یا آنرا از حالت استراحت بیرون می‌آورد. بیت More نشان می‌دهد که فرستنده باز هم فریمهائی برای ارسال به گیرنده آماده دارد. بیت W مشخص می‌کند که بدنهٔ فریم با استفاده از الگوریتم WEP (Wire Equivalent Privacy) (فصل نهم) رمزنگاری شده است. نهایتاً بیت O به گیرنده تفهیم می‌کند دنباله‌ای از فریمها که این بیت در آنها ۱ است باید الزاماً به ترتیب (و پشت سرهم) پردازش شوند.

فیلد دوم از فریم داده یعنی فیلد Duration، مشخص‌کنندهٔ آنست که ارسال فریم جاری و دریافت ACK آن، جمعاً چه زمانی کانال را به حالت اشغال در خواهد آورد. این فیلد که در فریمهای کنترلی نیز وجود دارد مشخص می‌نماید که ایستگاه‌های دیگر به چه نحوی باید مکانیزم NAV خود را مدیریت کنند. سرآیند فریم حاوی چهار آدرس است که همگی منطبق با قالب استانداردی هستند که توسط IEEE 802 تعریف شده است. بدیهی است که به دو فیلد آدرس مبداء و مقصد نیاز بوده ولیکن دوتای دیگر چه کاربردی دارند؟ به خاطر داشته باشید که یک فریم ممکن است از طریق ایستگاه ثابت به یک سلول وارد یا از سلول خارج شود. دو آدرس اضافی، برای تعیین ایستگاه ثابت مبداء و مقصد بکار می‌رود. (وقتی که ترافیک داده‌ها بین چند سلول در حال جریان است).

فیلد Sequence اجازه می‌دهد تا قطعات یک فریم شماره‌گذاری شوند. از شانزده بیت موجود در این فیلد ۱۲ بیت، هویت فریم را و ۴ بیت شمارهٔ قطعه را مشخص می‌کند. در فیلد Data داده‌های خام قرار می‌گیرد که می‌تواند حداکثر ۲۳۱۲ بایت باشد. در آخر نیز فیلد Checksum قرار گرفته که به منظور کشف خطا کاربرد دارد.

«فریم مدیریت» قالبی مشابه با «فریم داده» دارد با این تفاوت که این فریم یکی از آدرسهای ایستگاه ثابت را ندارد، زیرا فریمهای مدیریتی فقط محدود به یک سلول خاص هستند [و بین سلول‌های متفاوت مبادله نخواهند شد.] فریم کنترلی بازهم کوتاهتر هستند و فقط یک یا حداکثر دو فیلد آدرس دارند. این فریمها، فاقد فیلد داده و فیلد Sequence هستند. اطلاعات اساسی اینگونه فریمها، درون فیلد Subtype نهفته است که عموماً نوع RTS، CTS یا ACK را مشخص می‌کند.

٤-٤-٥ خدمات

استاندارد 802.11 بیان داشته که هر شبکه محلی بی‌سیم باید ۹ نوع خدمات عرضه نماید. این خدمات به دو رده تقسیم بندی شده‌اند. پنج نوع خدمات «**توزیعی**» و چهار نوع خدمات «**ایستگاهی**». خدمات توزیعی در خصوص مدیریت بر عضویت ایستگاههای درون سلول و تعامل با ایستگاه‌های خارج از سلول است. در مقابل، خدمات ایستگاهی صرفاً در خصوص فعالیتهای درون یک سلول واحد ارائه می‌شود.

پنج نوع خدمات توزیعی که توسط ایستگاه‌های ثابت عرضه می‌شوند ، در خصوص قابلیت تحرّک ایستگاه‌های همراه ، ورود و خروج آنها به سلولها و اتصال یا انفصال از ایستگاه ثابت کاربرد دارند. این خدمات عبارتند از:

۱. Association (**پیوستن به شبکه**): ایستگاه‌های متحرک از این سرویس بهره می‌گیرند تا خود را به ایستگاه ثابت متصل نمایند. بطور معمول زمانی که یک ایستگاه متحرک وارد محدودهٔ رادیوئی یک ایستگاه ثابت می‌شود با استفاده از این سرویس، هویت و قابلیتهای خود را معرفی می‌کند. قابلیتها عبارتند از نرخ ارسال داده‌ها (نرخ‌های متعددی که از آن حمایت می‌شود)، نیاز به خدمات متعدّد PCF (مثل سرکشی) و نیازمندیهای آن در خصوص مدیریت توان مصرفی. ایستگاه ثابت می‌تواند حضور ایستگاه متحرک را بپذیرد یا ردّ کند. اگر ایستگاه متحرک پذیرفته شد بایستی هویت خود را اثبات کند. [روشهای احراز هویت را در فصل نهم مطالعه کنید.]

۲. Disassociation (**ترک شبکه**): ممکن است ایستگاه متحرک یا ایستگاه ثابت، در هر زمان اراده کنند از یکدیگر جدا شوند و ارتباط خود را قطع نمایند. وقتی ایستگاهی بخواهد شبکه را ترک کند یا خاموش شود، از این سرویس بهره می‌گیرد اما ایستگاه ثابت نیز قبل از قطع موقّت ارتباط ممکن است از آن استفاده کند.

۳. Reassociation (**پیوستن مجدد**): یک ایستگاه متحرک می‌تواند با استفاده از این سرویس، ایستگاه ثابت خود را تغییر بدهد. این قابلیت زمانی کاربرد دارد که ایستگاه‌های متحرک بخواهند از یک سلول به سمت سلول دیگر حرکت نمایند. اگر از این امکان به موقع و صحیح استفاده شود در اثر این جابجائی، هیچ داده‌ای از دست نخواهد رفت.

۴. Distribution (**توزیع**): این سرویس مسیر ارسال فریمهای ارسالی به سوی ایستگاه ثابت را تعیین می‌نماید. اگر مقصد فریمها در همان محل ایستگاه ثابت واقع شده باشد فریم می‌تواند مستقیماً از طریق هوا ارسال شود. در غیر این صورت فریمها باید از طریق شبکهٔ سیمی (ارتباط سیمی بین ایستگاه‌های ثابت) به مقصد هدایت شود.

۵. Integration (**یکپارچگی**): اگر نیاز باشد فریمی به شبکه‌ای غیر از 802.11 (با ساختار آدرس و ساختار فریم متفاوت) ارسال شود، این سرویس می‌تواند وظیفه ترجمه و تبدیل قالب فریم (متناسب با شبکه مقصد) را برعهده بگیرد.

چهار سرویس باقیمانده در درون یک سلول بکار می‌آیند. (بعبارت دیگر در خصوص عملیات درون یک سلول واحد کاربرد دارند.) این سرویسها پس از پیوستن ایستگاه‌ها به یک ایستگاه ثابت [به عنوان عضوی از شبکه] مورد استفاده قرار گرفته و به عبارت زیر هستند:

۱. Authentication (**احراز هویت**): از آنجایی که در مخابرات بی‌سیم ایستگاه‌های غیرمجاز و بیگانه نیز می‌توانند ارسال و دریافت داشته باشند فلذا هر ایستگاه باید قبل از دریافت مجوز ارسال، هویت خود را اثبات کند. پس از آنکه یک ایستگاه متحرک به عنوان عضو ، به یک ایستگاه ثابت پیوست (یا بعبارت دیگر درون

سلول پذیرفته شد)، ایستگاه ثابت یک فریم خاص به نام «فریم چالش» (Challenge) برای او می‌فرستد تا ببیند آیا آن ایستگاه کلید سرّی (کلمه عبور) خود را می‌داند یا نه؟ ایستگاه، آگاهی خود از کلید سرّی را با رمز کردن فریم و بازگرداندن آن، اثبات می‌نماید. اگر نتیجه درست باشد، ایستگاه متحرک در درون سلول ثبت‌نام و عضو می‌شود. در استاندارد اولیه لازم نبود که ایستگاه ثابت نیز عضویت خود را احراز کند در حالیکه اصلاح این اشکال بزرگ در دست اجرا است.

۲. Deauthentication (**لغو حضور در شبکه**): وقتی یک ایستگاه که قبلاً احراز هویت (و عضو) شده بخواهد شبکه را ترک کند باید لغو حضور خود در شبکه را به اطلاع ایستگاه ثابت برساند. پس از لغو حضور و ترک شبکه، ایستگاه دیگر نمی‌تواند از شبکه بهره بگیرد.

۳. Privacy (**محرمانه نگاه داشتن اطلاعات**): برای محرمانه ماندن اطلاعاتی که از طریق شبکهٔ بی‌سیم مبادله می‌شود، بایستی رمز شوند. این کار از طریق رمزنگاری و رمز گشائی میّسر است. الگوریتم رمزنگاری مورد استفاده روش RC4 می‌باشد که توسط رونالد ری‌وست در دانشگاه MIT ابداع شده است.

۴. Delivery (**تحویل**): انتقال داده‌ها کل آن هدفی است که در پی آن هستیم لذا در 802.11 روشی برای ارسال و دریافت داده‌ها ارائه شده است. از آنجایی که 802.11 مبتنی بر مدل شبکهٔ اترنت است و مبادله اطلاعات در اترنت تضمین صددرصد ندارد، 802.11 نیز مبادله مطمئن داده‌ها را تضمین نمی‌کند. لایه‌های بالاتر باید در خصوص کشف و تصحیح خطا کاری انجام بدهند.

در استاندارد 802.11، هر سلول پارامترهائی دارد که می‌توان آنها را بررسی و در صورت نیاز تنظیم کرد. این پارامترها در خصوص عملیات رمزنگاری، نرخ ارسال، بازه‌های زمانی [زمان‌های انقضای مهلت یا Timeouts]، تناوب ارسال فریمهای Beacon ونظایر اینها، تعریف شده‌اند.

شبکه‌های محلی بی‌سیم 802.11، در حال ورود به عرصه ادارات، فرودگاه‌ها، هتلها، رستورانها و محوطه‌های دانشگاهی هستند و انتظار می‌رود رشد سریع و چشمگیری داشته باشند. برای آنکه در خصوص استفادهٔ گسترده از 802.11، اطلاعات و تجاربی بدست بیاورید به مرجع (Hills, 2001) مراجعه کنید.

۵ـ۴ بی‌سیم با باند گسترده

بیش از اندازه به درون (شبکه‌های LAN) پرداخته‌ایم. حال اجازه بدهید پا را فراتر گذاشته و ببینیم آیا در خارج از دنیای LAN هم شبکه‌های جالبی وجود دارند. با رفع موانع قانونی از مقررات حاکم بر عرضهٔ خدمات تلفن، در بسیاری از کشورها شرکتهای مخابراتی رقیب اجازه یافتند تا به ارائه خدمات صوت و خدمات اینترنت پرسرعت بپردازند. امروزه نیاز عمومی در این خصوص بسیار بالا و پررونق است. برای چنین شرکتهایی مسئله اساسی آنست که کشیدن کابلهای فیبرنوری، کابل کوآکسیال یا سیمهای زوجی (از نوع CAT 5) تا درب در میلیونها خانه و محل کار، بسیار گران و نامعقول به نظر می‌رسد. پس این رقبای تجاری برای عرضه خدمات ارزان، چه باید می‌کردند؟

پاسخ به این نیاز «**شبکهٔ بی‌سیم‌باند گسترده**» است. برافراشتن یک آنتن بزرگ برروی قله یا یک تپه و نصب آنتنهائی برروی پشت بام مشتریان که به سمت آنتن اصلی جهت‌گیری شده، بسیار ارزان‌تر از حفر زمین و کابل‌کشی است. بهمین دلیل شرکتهای مخابرات راه دور بیشتر گرایش دارند که برای عرضه خدمات اینترنت، ارسال صدا و ارائه نمایشهای ویدیوئی راه دور، از سیستمهای چندگیگابیتی بی‌سیم بهره بگیرند. همانگونه که در شکل ۲ـ۳۰ ملاحظه نمودید، LMDS به همین منظور ابداع شد ولیکن تاکنون هر یک از شرکت‌های عرضه‌کنندهٔ

LMDS ، به سلیقهٔ خود سیستمی را طراحی و نصب کرده است. فقدان یک استاندارد واحد، بدین معناست که سخت‌افزار و نرم‌افزار آنها نمی‌تواند بطور انبوه تولید شود و طبعاً قیمتها بالا و استقبال عمومی از آنها کم خواهد بود.

نهایتاً بسیاری از دست اندرکاران صنعت بدین نتیجه رسیدند که داشتن یک استاندارد برای بی‌سیم باند گسترده [با نرخ ارسال بسیار بالا] یکی از مولفه‌های اصلی و حلقهٔ مفقوده در کار آنهاست، لذا از IEEE خواستند تا برای تدوین چنین استانداردی ، کمیته‌ای با حضور دست‌اندرکاران شرکتهای مهم و مراکز دانشگاهی، تشکیل بدهد. اولین عدد اختصاصی داده نشده در ردیف شماره‌های 802.x ، ۱۶ بود و به همین دلیل استاندارد ، با عنوان IEEE 802.16 معرفی شد. کار کمیته در ژولای ۱۹۹۹ شروع و استاندارد نهائی در آوریل ۲۰۰۲ به تائید رسید. نام رسمی استاندارد «**واسط هوائی برای سیستمهای بی‌سیم غیرمتحرک با پهنای باند وسیع**»[1] انتخاب شده ولیکن برخی از افراد ترجیح می‌دهند آنرا «شبکه بین‌شهری بی‌سیم» یا «حلقهٔ بی‌سیم» بنامند. ما تمام این واژه‌ها را معادل یکدیگر فرض خواهیم کرد.

همانند بسیاری از استانداردهای دیگر سری ۸۰۲، استاندارد 802.16 نیز شدیداً تحت تاثیر مدل OSI بوده و این تاثیر در (زیر) لایه‌ها، اصطلاحات، سرویسهای پایه و موارد دیگر بخوبی محسوس است. متاسفانه این استاندارد نیز شبیه به OSI پیچیده شده است. در بخش بعدی بطور مختصر نکات و ویژگیهای برجستهٔ 802.16 را برخواهیم شمرد ولی این توضیحات به هیچ وجه کامل نبوده و جزئیات آن ناگفته خواهد ماند.

۱.۵.٤ مقایسهٔ 802.11 با 802.16

در همان ابتدا ممکن است بدین نکته بیندیشید که چرا استاندارد جدیدی ابداع شد؟ چرا از 802.11 استفاده نشد؟ دلائل متعدد و محکمی برای عدم استفاده از 802.11 وجود دارد. در اصل 802.11 و 802.16 نیازهای متفاوتی را برآورده می‌کنند. قبل از پرداختن به فناوری 802.16 شاید چند کلمه‌ای بحث در خصوص دلائل نیاز به استاندارد جدید، خالی از لطف نباشد.

محیطی که 802.11 و 802.16 در آن، عمل می‌کنند از چند منظر شبیه به هم هستند: در اصل هر دوی این استانداردها بدان جهت طراحی شده‌اند که ارتباط بی‌سیم با پهنای باند بسیار بالا را میسّر نمایند. ولیکن این دو شبکه از جهات بسیار مهمی با هم متفاوتند. اصلی‌ترین تفاوت آنست که 802.16 برای ارائه خدمات به ساختمانها طراحی شده است و طبعاً ساختمانها حرکت نمی‌کنند!!! ساختمانها تغییر سلول نمی‌دهند! بخش اعظم عملیات 802.11 با مسائل ناشی از متحرک بودن ایستگاه‌ها سروکار دارد و چنین مسائلی در 802.16 ، محلّی از اعراب ندارد. گذشته از آن، در ساختمانها ممکن است بیش از یک کامپیوتر وجود داشته باشد و پیچیدگیهای چنین محیطی در جائی که ایستگاه نهائی یک کامپیوتر کیفی واحد است، بروز نخواهد کرد. از آنجایی که مالکین ساختمان معمولاً آمادگی پرداخت پول بیشتری در مقایسه با صاحب یک کامپیوتر کیفی دارند لذا می‌توان خدمات ارتباط رادیوئی بهتری در اختیارشان گذاشت. این تفاوت بدین معناست که در 802.16 می‌توان ارتباط دوطرفهٔ همزمان (Full Duplex) داشت در حالیکه برای پائین نگاه داشتن هزینهٔ ارتباط رادیوئی در 802.11 از آن اجتناب شده است. [ارتباط 802.11 دو طرفهٔ غیر همزمان ـ Half Duplex ـ و با برد بسیار کم می‌باشد.]

از آنجایی که 802.16 در محدودهٔ بخشی از یک شهر به اجرا در می‌آید، فواصل [بین نقاط] می‌تواند تا چندین کیلومتر باشد و این موضوع بدان معناست که توان مورد نیاز ایستگاه ثابت می‌تواند بسته به موقعیت و فاصله‌ها متغیر باشد. این تفاوتها و تغییرها برروی نسبت سیگنال به نویز (SNR) تاثیر گذاشته و در نتیجه، باید به اجبار از

---

۱. Air Interface for Fixed Broadband Wireless Access Systems

چندین روش مدولاسیون استفاده شود. همچنین مخابرهٔ آزادانه در سطح یک شهر مبیّن آنست که تمهیدات امنیتی و حفظ حریم افراد، نیازی بنیادی و اجتناب‌ناپذیر است.

به علاوه، در مقایسه با سلول‌های معمول 802.11 ، در اینجا سلولها می‌توانند کاربران بسیار زیادتری را در بر بگیرند و این کاربران نیز توقع پهنای باند بیشتری نسبت به کاربران 802.11 دارند. گذشته از آن به ندرت اتفاق می‌افتد که شرکتی از پنجاه کارمند خود دعوت کند تا برای مشاهدهٔ اشباع شدن [و از کار افتادن 802.11] دور هم جمع شده و پنجاه فیلم مجزا تماشا کنند!! [در حالیکه در یک ساختمان مسکونی احتمال دارد پنجاه نفر بدیدن فیلم از طریق شبکه مشغول باشند.] به همین دلیل به پهنای وسیعتری از طیف فرکانسی موجود در باند ISM نیاز است و طبعاً 802.16 مجبور است در محدودهٔ فرکانسی ۱۰ تا ۶۶گیگاهرتزی عمل کند؛ یعنی تنها محدودهٔ استفاده نشده از طیف فرکانس که هنوز موجود است.

ولیکن این امواج با طول موج میلی‌متری نسبت به امواج با طول موج بیشتر در باند ISM ، ویژگی‌های فیزیکی کاملاً متفاوتی دارند و در نتیجه به لایهٔ فیزیکی کاملاً متفاوتی نیاز است. یکی از ویژگی‌های امواج میلی‌متری آنست که توسط آب شدیداً جذب می‌شوند. (بالاخص باران، در برخی از مواقع برف، تگرگ و متاسفانه مه غلیظ) در نتیجه مدیریت خطا بسیار با اهمیت‌تر از محیط‌های داخلی [مثل محیط LAN] است. امواج میلی‌متری می‌توانند به خط مستقیم متمرکز و منتشر شوند (در حالیکه در 802.11 انتشار امواج در همهٔ جهات است) فلذا گزینه‌ها و تمهیدات پیش‌بینی شده در ارتباط با انتشار چندمسیره (Multipath) در 802.16 محلّی از اعراب ندارد.

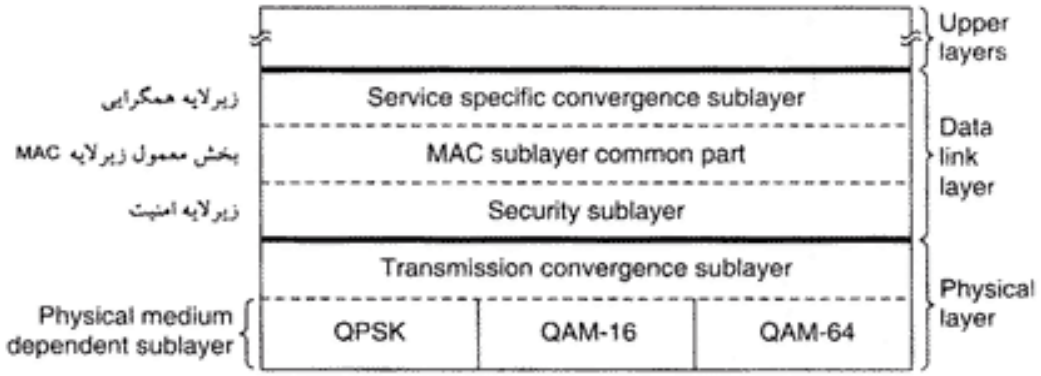
مورد دیگر مربوط به «**کیفیت خدمات**» (QoS) است. اگر چه 802.11 برای ترافیک بی‌درنگ پشتیبانیهائی به عمل آورده (در حالت PCF) ولیکن حقیقتاً برای کاربردهای تلفنی و عملیات چندرسانه‌ای سنگین و دائم طراحی نشده است. در مقابل، از 802.16 انتظار می‌رود که کاملاً از چنین کاربردهائی پشتیبانی نماید زیرا برای استفاده در محیط‌های مسکونی و اداری مدّ نظر بوده است.

کوتاه سخن آنکه 802.11 طراحی شده تا یک اترنت متحرک باشد در حالیکه 802.16 طراحی شده تا شبیه به تلویزیون کابلی، ثابت ولی بی‌سیم عمل نماید. این تفاوتها آنقدر بنیادیست که استانداردهای حاصل اختلاف فراوانی دارند و در هر یک سعی در بهینه‌سازی و حل و فصل نیازهای متفاوتی شده است.

مقایسه‌ای بسیار کوتاه با سیستم تلفن سلولی [تلفن همراه] نیز خالی از فایده نیست. وقتی در مورد تلفن‌های همراه صحبت می‌کنیم روی سخن با ایستگاه‌هائی همراه با محوریت ارسال صوت، توان مصرفی پائین و پهنای باند باریک است که از امواج مایکروویو با طول موج متوسط بهره گرفته‌اند. هیچ‌کس یک فیلم دو ساعته و با کیفیت بالا را برروی گوشی موبایل GSM خود تماشا نمی‌کند (البته فعلاً)!! حتی UMTS نیز امیدی به تغییر چنین وضعیتی ندارد. [امید به افزایش چشمگیر پهنای باند در GSM] در عبارتی کوتاه در دنیای شبکه‌های بین شهری بی‌سیم (Wireless MAN) تقاضا برای پهنای باند بسیار بیشتر از دنیای تلفن همراه است و طبعاً به سیستمهائی کاملاً متفاوت نیاز می‌باشد. اینکه آیا 802.16 می‌تواند در آینده برای ابزارهای متحرک و همراه نیز به کارگرفته شود سوال جالبی است: این شبکه برای چنین محیطی بهینه نشده ولیکن شاید بشود! فعلاً تمرکز آن برروی شبکهٔ دادهٔ بی‌سیم و غیرمتحرک است.

۲-۵-٤ پشتهٔ پروتکلی 802.16

شکل ۴-۳۱ پشتهٔ پروتکلی 802.16 را نشان می‌دهد. ساختار کلّی این پشته، شبیه به شبکه‌های دیگر 802 است ولیکن تعداد بیشتری زیرلایه دارد. پائینترین زیرلایه با انتقال فیزیکی بیتها بر روی کانال سروکار دارد و در آن از رادیوی باند باریک و روش‌های معمول مدولاسیون، بهره گرفته شده است. برروی «لایهٔ فیزیکی انتقال» ، زیرلایهٔ «**همگرائی**» (Convergence Sublayer) قرار گرفته تا تکنولوژیهای متفاوت به کار رفته در زیر را از دید لایهٔ پیوند



شکل ۴ـ۳۱. پشتهٔ پروتکلی 802.16.

داده‌ها مخفی نگاه دارد. در حقیقت، 802.11 نیز چیزی شبیه به همین زیرلایه را دارد ولی کمیتهٔ مربوطه ، آنرا با استفاده از اسامی رایج در مدل OSI تعریف نکرده است.

اگرچه در شکل ۴ـ۳۱ نشان نداده‌ایم ولی در آینده دو پروتکل جدید به لایهٔ فیزیکی اضافه می‌شود: (۱) استاندارد 802.16a که از روش مدولاسیون OFDM در باند ۲ تا ۱۱ گیگاهرتز پشتیبانی می‌نماید. (۲) استاندارد 802.16b که در باند ۵ گیگاهرتز ISM عمل می‌کند. در هر دوی اینها تلاش شده تا 802.16 به 802.11 نزدیکتر شود.

لایهٔ پیوند داده متشکل از سه زیرلایه است: زیرلایهٔ پائینی با امنیت و محرمانه نگاه داشتن اطلاعات سروکار دارد که برای شبکه‌های عمومی در محیطهای باز بسیار حیاتی‌تر از شبکه‌های خصوصی بسته مثل اترنت است. این زیرلایه عملیات رمزنگاری، رمزگشائی و مدیریت کلیدها را برعهده دارد.

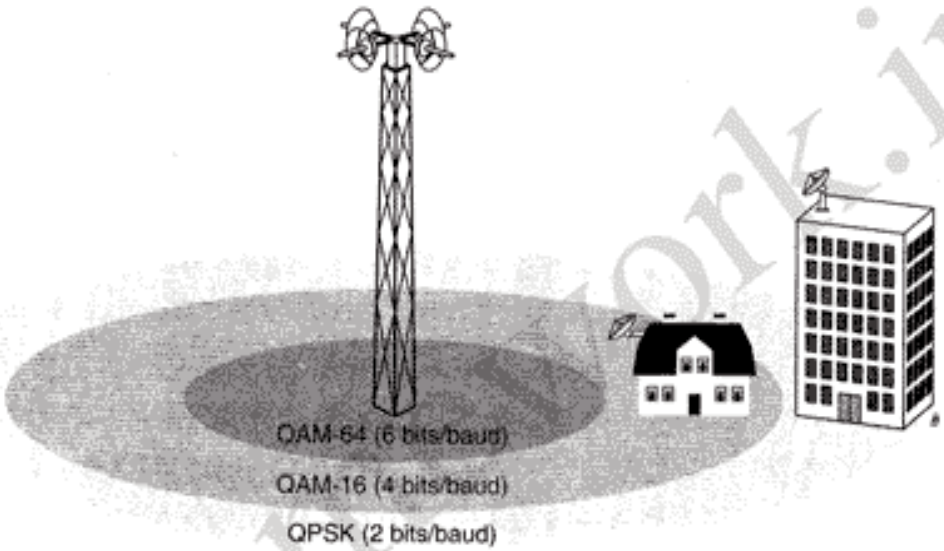
سپس بخش مشترک از زیرلایهٔ MAC قرار می‌گیرد. این لایه همان نقطه‌ای است که پروتکلهای اساسی مثل پروتکلهای مدیریت کانال در بر می‌گیرد. در این مدل مبنا آنست که ایستگاه ثابت ، سیستم را کنترل می‌کند. این ایستگاه می‌تواند «جریان اطلاعات از ایستگاه ثابت به مشترکین» (که اصطلاحاً downstream نام دارد) را به صورت کاملاً متفاوت و منمایز زمان‌بندی نماید و نیز نقش بسیار مهمی در مدیریت «جریان اطلاعات از مشترکین به ایستگاه ثابت» (Upstream) ایفا می‌کند. یکی از ویژگیهای نامتعارف در زیرلایهٔ MAC آنست که برخلاف شبکه‌های دیگر 802 ، این استاندارد کلاً اتصال‌گرا (Connection Oriented) است، تا بتواند «کیفیت خدمات» (QoS) را برای ارتباطات تلفنی و سایر کاربردهای چندرسانه‌ای تضمین نماید.

زیرلایهٔ **«همگرائی خاص سرویس‌دهی»** (Service Specific Convergence Sublayer) به جای «زیرلایهٔ لینک منطقی» (یعنی زیرلایهٔ معمول LLC) قرار گرفته است. عملکرد این زیرلایه ایجاد واسطی متناسب با لایهٔ شبکه است. پیچیدگی این زیرلایه از آنجا نشأت می‌گیرد که 802.16 به نحوی طراحی شده تا بتواند با پروتکلهای دیتاگرام (مثل PPP ، IP و اترنت) و همچنین شبکهٔ اتصال‌گرای ATM قابل جمع باشد و با آنها کار کند. مشکل آنجاست که پروتکلهای «مبتنی بر بسته» بدون‌اتصال (Connectionless) هستند در حالیکه ATM اتصال‌گراست. این بدین معناست که هر اتصال ATM بایستی به یک اتصال در 802.16 نگاشته شود که اصولاً کار ساده و سرراستی است. ولی سوال این است که یک بستهٔ IP ورودی باید برروی کدامین اتصال 802.16 نگاشته شود؟ این مشکل در همین زیرلایه حلّ خواهد شد.

۴ـ۵ـ۳ لایهٔ فیزیکی در 802.16

بالا اشاره شد که در بی‌سیم با پهنای باند وسیع، به جس گسترده‌تری از طیف فرکانسی نیاز است و تنها محل

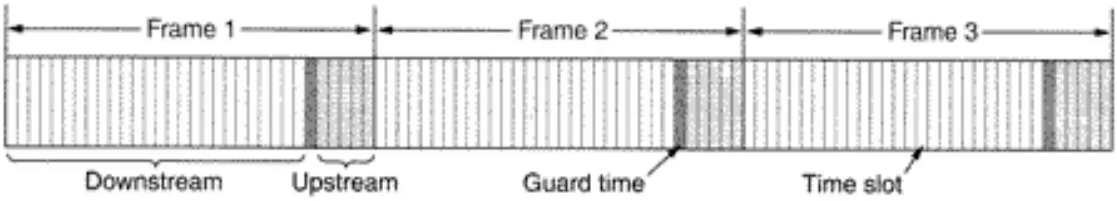
دسترسی به چنین وسعتی محدودهٔ ۱۰ تا ۶۶ گیگاهرتزی است. این امواج با طول‌موج میلی‌متری ویژگیهای جالبی دارند که امواج مایکروویو با طول موج بلندتر ندارند: آنها برخلاف صوت و مشابه با نور به خط مستقیم سیر می‌کنند. در نتیجه، ایستگاه ثابت بایستی چندین آنتن داشته باشد و هر یک از آنها بسوی قطاع خاصی از مناطق پیرامون خود نشانه رفته باشد. (شکل ۴-۳۲) هر قطاع ، کاربران خاصّ خود را دارد و مستقل از قطاعهای همجوار خود است؛ چنین ساختاری برای رادیوی سلولی صادق نیست چرا که آنتنهای آنها «همه‌جهته» (Omnidirectional) هستند.



شکل ۴-۳۲. محیط انتقال در 802.16

از آنجایی که در باند امواج میلی‌متری توان سیگنال براساس فاصله از ایستگاه ثابت، شدیداً کاهش می‌یابد لذ نسبت سیگنال به نویز (SNR) نیز برحسب فاصله افت خواهد داشت. به همین دلیل 802.16 بسته به فاصلهٔ یک مشترک از ایستگاه ثابت، سه روش مدولاسیون متفاوت را به کارگرفته است. برای مشترکین نزدیک، از روش QAM-64 با مشخصهٔ 6 bits/baud استفاده شده است. برای مشترکین با فاصلهٔ متوسط از QPSK با مشخصهٔ 2 bits/baud استفاده می‌شود. برای مشترکین دوردست ، روش QPSK با مشخصه 2 bits/boud به کار می‌رود. برای مثال به ازای پهنای معمول ۲۵ مگاهرتزی از طیف فرکانس، پهنای باند روش QAM-64 نرخ ۱۵۰ مگابیت بر ثانیه. روش QAM-16 نرخ ۱۰۰ مگابیت بر ثانیه و QPSK نرخ ۵۰ مگابیت بر ثانیه را در اختیار قرار می‌دهد. به عبارت دیگر هر چه مشترکین از ایستگاه ثابت دورتر باشند، نرخ ارسال پائینتر خواهد بود (شبیه به آنچه که در خصوص ADSL در شکل ۲-۲۷ دیدید) در شکل ۲-۲۵ نمودار فضائی این سه مدولاسیون ، نمایش داده شده است.

با توجه به آنکه هدف اصلی ایجاد سیستمی با باند وسیع بوده و با در نظر داشتن محدودیتهای فیزیکی. طراحان 802.16 کوشیدند تا از طیف موجود به نحو بهینه استفاده کنند. آنها به روش به کاررفته در GSM و DAMPS علاقه‌ای نداشتند. هر دوی این سیستمها از فرکانسهای متفاوت ولی در باند مشابهی برای ارسال جریان ترافیک رو به بالا و رو به پائین (Upstream/Downstream) بهره می‌گیرند. برای صوت شاید ترافیک در اکثر بخشها متقارن باشد [بعبارت دیگر برای صوت میزان ارسال و دریافت حدوداً به یک اندازه است] ولیکن برای دسترسی به اینترنت غالباً حجم ترافیک دریافتی بیشتر از ترافیک ارسالی است. در نتیجه 802.16 روشی منعطفتر برای تخصیص پهنای باند ارائه کرده و از دو روش FDD (Frequency Division Duplexing) و TDD (Time Division Duplexing) بهره گرفته است. روش دوّم یعنی TDD . در شکل ۴-۳۳ نشان داده شده



شکل ۴-۳۳. فریمها و برشهای زمانی در روش TDD (Time Division Duplexing).

است. در اینجا ایستگاه ثابت بطور متناوب فریمهائی را منتشر می‌کند. هر فریم شامل تعدادی برش زمانی مستقل (Time Slot) است. برشهای ابتدائی هر فریم، برای ارسال ترافیک رویه‌پائین [یعنی از ایستگاه ثابت به کاربر] در نظر گرفته شده است. پس از آن «**زمان مراقبت**» (Guard Time) فرا می‌رسد که به ایستگاه‌ها مهلت می‌دهد تا جهت ارسال و دریافت خود را تغییر بدهند. در آخر برشهائی را برای ارسال ترافیک رو به بالا [از کاربر به ایستگاه ثابت] خواهیم داشت. تعداد برشهای زمانی را که به ارسال در هر یک از جهات، اختصاص می‌یابد، می‌توان به صورت پویا تغییر داد تا پهنای باند در هر جهت، با حجم ترافیک تطبیق داشته باشد.

ترافیک رویه‌پائین توسط ایستگاه ثابت در درون برشهای زمانی نگاشته می‌شود. ایستگاه ثابت بطور کامل بر این جهت از جریان، کنترل دارد. ترافیک رویه‌بالا [که توسط کاربران تولید می‌شود] با پیچیدگی بیشتری مواجه بوده و میزان آن به کیفیت خدمات (QoS) مورد نیاز بستگی دارد. در ادامه وقتی به تشریح زیرلایهٔ MAC پرداختیم به روش تخصیص برشهای زمانی هم خواهیم رسید.

ویژگی جالب دیگر در لایهٔ فیزیکی، توانائی آن در ارسال متوالی و پشت سرهم فریمهای MAC، در قالب یک انتقال فیزیکی واحد است. این ویژگی سربار ناشی از بیتهای آغازین (Preamble) و سرآیند لازم در لایهٔ فیزیکی را کاهش داده و در نتیجه بازده مفید طیف را افزایش خواهد داد.

نکتهٔ قابل توجه دیگر، استفاده از **کدهای همینگ** (Hamming) به منظور تصحیح مستقیم خطا در لایهٔ فیزیکی است.[1] تقریباً در تمام شبکه‌ها، به کدهای کشف خطا بسنده می‌شود و هرگاه فریم دریافتی دارای خطا باشد ارسال مجدّد صورت می‌گیرد، ولی از آنجاییکه در محیط‌های باز و در نرخ ارسال بالا، احتمال بروز خطا در حین انتقال، خیلی بیشتر است لذا گذشته از عملیات کشف خطا (که در لایه‌های بالاتر انجام می‌شود) در لایهٔ فیزیکی نیز عملیات تصحیح خطا صورت می‌گیرد. تاثیر نهائی تصحیح خطا آنست که کانال بهتر از آنی که هست به نظر می‌رسد. (بدلیل مشابه اگر چه CD-ROMها قابل‌اعتماد به نظر می‌رسند ولیکن این اعتماد، حاصل از تخصیص بیش از نیمی از بیتها به تصحیح خطا در لایهٔ فیزیکی است.) [به عبارت دیگر در هر CD، بیش از نیمی از بیتهای سطح دیسک فقط به منظور عملیات کشف و تصحیح خطا ذخیره شده‌اند!]

٤-٥-٤ پروتکل زیرلایهٔ MAC در 802.16

همانگونه که در شکل ۴-۳۱ دیدیم، در 802.16، لایهٔ پیوند داده‌ها به سه زیرلایه تقسیم می‌شود. از آنجایی که ما تا فصل هشتم به مطالعه رمزنگاری نخواهیم پرداخت لذا تشریح عملکرد «زیرلایهٔ امنیت» در اینجا سخت است. به همین مقدار بسنده می‌کنیم که برای سرّی نگاه داشتن داده‌های ارسالی از رمزنگاری در سطح لایهٔ فیزیکی و به صورت بی‌درنگ، بهره گرفته شده است. صرفاً بخش دادهٔ هر فریم رمزنگاری می‌شود و سرآیند آن (Header) رمز نخواهد شد. این خصوصیّت بدین معناست که یک جاسوس قادر است بفهمد چه کسی با چه کسی محاوره می‌کند ولی قادر به فهم آنچه با یکدیگر می‌گویند نیست.

---

۱. Forward Error Correction

اگر با رمزنگاری آشنائی قبلی داشته باشید در حدّ یک پارگراف، زیرلایهٔ امنیت را بررسی می‌کنیم ولیکن در صورت عدم آشنائی با رمزنگاری، پارگراف بعدی چیزی به دانش شما نخواهد افزود. (می‌توانید این پارگراف را پس از اتمام فصل ۸ مجدداً مطالعه نمائید.)

زمانی که یک مشترک به ایستگاه ثابت متصل می‌شود، دوبه‌دو یکدیگر را با استفاده از روش «رمزنگاری RSA» و مبتنی بر «گواهینامه‌های X.509» احراز هویت می‌کنند. بخش دادهٔ فریمهای آنها با استفاده از یک سیستم رمزنگاری متقارن رمز می‌شود که این سیستم یا مبتنی بر «DES با زنجیره‌سازی بلوکها» است و یا از روش «DES سه‌گانه با دو کلید» استفاده می‌شود. احتمالاً به زودی روش AES (Rijndael) نیز به آن اضافه می‌شود. عملیات بررسی صحّت داده‌ها نیز با استفاده از SHA-1 انجام می‌گیرد. تا اینجا چندان بد نبود، نظر شما چیست؟!!

حال اجازه بدهید نگاهی به بخش مشترک از زیرلایهٔ MAC بیندازیم. فریمهای MAC ، تعداد مشخصی از برشهای زمانی لایهٔ فیزیکی را اشغال می‌کنند. هر فریم از چندین فریم کوچکتر (Subframe) تشکیل شده که دو تای ابتدائی آن به ترتیب برای نگاشت و حمل ترافیک روبه‌بالا (Upstream) و روبه‌پائین (Downstream) در نظر گرفته شده است. ساختار این نگاشت (یعنی درج داده‌ها درون برشهای زمانی) بگونه‌ای است که مشخص می‌کند چه چیزی در یک برش زمانی درج شده و کدامیک از برشهای زمانی آزاد هستند. نگاشت ترافیک روبه‌پائین نیز شامل پارامترهای مختلف سیستمی است تا شروع فعالیت یک ایستگاه جدید را به اطلاع ایستگاه ثابت برساند.

عملکرد کانال روبه‌پائین نسبتاً ساده و سر راست است. ایستگاه ثابت بسادگی تصمیم می‌گیرد که چه چیزی را در کدام فریم کوچک (Subframe) قرار بدهد. عملکرد کانال رو به بالا پیچیده‌تر است زیرا در اینجا مشترکین رقیب و ناهماهنگ، برای در اختیار گرفتن آن با یکدیگر رقابت می‌کنند. روش تخصیص کانال به مواردی در خصوص «کیفیت خدمات» (QoS) بستگی دارد. در 802.16 چهار رده از خدمات، به ترتیب ذیل تعریف شده‌اند:

۱. **خدمات با نرخ ارسال ثابت** (Constant Bit Rate Service)
۲. **خدمات بی‌درنگ با نرخ ارسال متغیر** (Real-time Variable Bit Rate)
۳. **خدمات غیر بی‌درنگ با نرخ ارسال متغیر** (NonReal-time Variable bit Rate)
۴. **خدمات مبتنی بر «حداکثر تلاش»** (Best Effort Service)

تمام خدمات عرضه شده در 802.16 اتصال‌گرا هستند و به هر اتصال[1] فقط یکی از رده‌های خدمات فوق تعلّق می‌گیرد و نوع خدمات نیز در زمان برقراری اتصال تعیین می‌شود. این طراحی بسیار متفاوت از 802.11 یا اترنت است که در آنها هیچ اتصالی در زیرلایهٔ MAC ایجاد نمی‌شود.

«خدمات با نرخ ارسال ثابت» برای انتقال سیگنال صوتی غیرفشرده همانند یک کانال T1 مدّ نظر بوده است. به این خدمات از آن جهت نیاز است که بتوان حجم معینی از داده‌ها را در زمان مشخصی ارسال کرد. برای رسیدن به این هدف، به هر اتصال از این نوع، تعداد ثابت و معینی برش زمانی (در هر فریم[2]) اختصاص داده می‌شود. هرگاه پهنای باند لازم اختصاص داده شود این برشهای زمانی بطور خودکار و بدون نیاز به درخواست بعدی در اختیار خواهد بود.

«خدمات بی‌درنگ با نرخ ارسال متغیر» برای کاربردهای چندرسانه‌ای فشرده‌شده و هرگونه عملیات بی‌درنگ که در آنها مقدار پهنای باند مورد نیاز، متغیر است سودمند خواهد بود. این کار بدین نحو انجام می‌شود که ایستگاه ثابت در فواصل زمانی مشخص به مشترک خود سرکشی کرده و از او در خصوص میزان پهنای باند مورد نیازش

---

۱. اتصال را یک ارتباط منطقی و هماهنگ شده بین دو نقطه در شبکه در نظر بگیرید. -م

۲. معنای واژهٔ فریم در اینجا با معنای متعارف آن تفاوت دارد. معنای فریم 802.16 در شکل ۴-۳۳ مشخص است. -م

سوال می‌کند.

«**خدمات غیربی‌درنگ با نرخ ارسال متغیر**» برای حجم ترافیک سنگینی که بی‌درنگ نیست (همانند انتقال فایل‌های طولانی) در نظر گرفته شده است. برای ارائهٔ چنین خدماتی، اغلب ایستگاه ثابت به مشترک خود سرکشی می‌کند ولیکن فواصل زمانی سرکشی به مشترک، قطعی و منظم نیست. یک مشترک که از نرخ ثابت بهره می‌گیرد می‌تواند یک بیت خاصّ را در یکی از فریمهای خود تنظیم کرده و تقاضای سرکشی بدهد تا بتواند ترافیک اضافی (با نرخ متغیر) ارسال نماید.

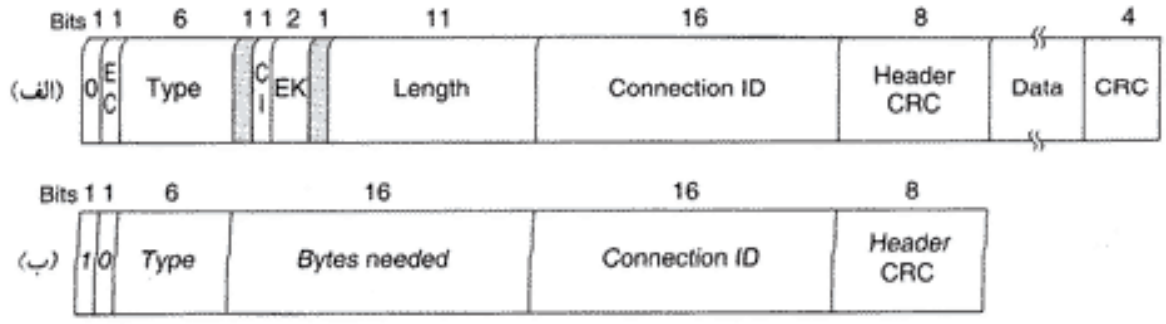
اگر ایستگاهی k بار متوالی سرکشی شود و پاسخی ندهد، ایستگاه ثابت او را در یک «گروه چندپخشی» (Multicast Group) قرار داده و از آن به بعد بصورت اختصاصی سرکشی نخواهد شد. در عوض وقتی به یک «گروه چندپخشی» سرکشی می‌شود هر کدام از ایستگاه‌های گروه می‌توانند پاسخ بدهند و برای دریافت خدمات رقابت کنند. بدین ترتیب ایستگاه‌هائی که ترافیک ناچیزی دارند زمان باارزش سرکشی را هدر نخواهند داد.

خدمات مبتنی بر «**بهترین تلاش**» در موارد متفاوتی کاربرد دارد. در اینجا هیچ سرکشی انجام نمی‌گیرد و هر مشترک برای دریافت این خدمات باید با مشترکین دیگر (که آنها نیز در پی خدمات مبتنی بر بهترین تلاش هستند) رقابت کند. تقاضای پهنای باند با علامتگذاری در یکی از برشهای زمانی ترافیک روبه‌بالا که برای رقابت تدارک دیده شده انجام می‌گیرد. اگر تقاضا به صورت موفقیت‌آمیز اعلان شود، این موفقیت در نگاشت ترافیک روبه‌پائین [فریم مربوطه به Downstream] مشخص خواهد شد. اگر تقاضا موفق نبود، مشترک بایستی مجدداً تلاش کند. برای آنکه تصادمها حداقل شود از الگوریتم عقب‌گرد نمایی در اترنت بهره گرفته شده است.

استاندارد 802.16 ، دو نوع تخصیص پهنای باند تدارک دیده است: تخصیص پهنای باند به ازای هر ایستگاه و تخصیص پهنای باند به ازای هر اتصال (per-connection) . در روش اول، ایستگاه نصب شده در یک ساختمان، کلیهٔ تقاضاهای کاربران را به صورت یکجا جمع کرده و به نیابت از همهٔ آنها تقاضای پهنای باند می‌کند و اگر توانست پهنای باند درخواستی را بدست بیاورد، این پهنای باند را به تناسب بین کاربران تقسیم می‌کند. در روش دوّم، ایستگاه ثابت هر اتصال را مستقلاً مدیریت می‌کند.

٤-٥-٥ ساختار فریم در 802.16

تمام فریمهای MAC با یک سرآیند عمومی شروع می‌شوند. به نحوی که در شکل ۴-۳۴ می‌بینید پس از سرآیند ، بخشی اختیاری جهت حمل داده و کد اختیاری کشف خطا (CRC) قرار گرفته است. در فریمهای کنترلی به بخش داده (Payload) نیازی نیست (مثلاً در فریمهایی که تقاضای برش زمانی می‌دهند). کد کشف خطا (Checksum) نیز اختیاری است زیرا در لایهٔ فیزیکی ، عملیات تصحیح خطا انجام می‌گیرد و همچنین قرار نیست فریمهای بی‌درنگ [در صورت خراب شدن] از نو ارسال شوند. اگر قرار نباشد که فریمی از نو ارسال شود چرا با افزودن کد کشف خطا، خود را به زحمت بیندازیم.



شکل ۴-۳۴. (الف) قالب عمومی فریم (ب) فریم تقاضای پهنای باند.

فیلدهای سرآیند شکل ۴-۳۴-الف را به اختصار معرفی می‌کنیم: بیت EC مشخص می‌کند که آیا بخش داده، رمزنگاری شده است؟ فیلد Type ، نوع فریم را تعیین می‌کند، بالاخص آنکه آیا داده‌ها به صورت مجموعه‌ای از قطعات یا آنکه به صورت یکجا ارسال می‌شود. فیلد CI مشخص می‌کند که آیا فیلد کشف خطا (Checksum) در پایان فریم وجود دارد یا خیر. فیلد EK مشخص‌کننده آنست که کدامیک از کلیدهای رمزنگاری به کارگرفته شده است. [بشرط آنکه رمزنگاری انجام و چندین کلید تعریف شده باشد]. فیلد Length طول کلّ فریم را با احتساب سرآیند تعیین می‌کند. فیلد Connection Identifier مشخص می‌کند که فریم متعلق به کدام «اتصال» است. در آخر، فیلد Header CRC در برگیرندهٔ کد کشف خطائی است که فقط از بخش سرآیند و با استفاده از چندجمله‌ای x^8+x^2+x+1 استخراج می‌شود.

نوع دوّم سرآیند که فقط برای فریمهائی که پهنای باند تقاضا می‌دهند تعریف شده، در شکل ۴-۳۴-ب نشان داده شده است. این سرآیند به جای بیت صفر با بیت ۱ شروع شده ولیکن ترکیب کلّی آن با ترکیب فریم عمومی [شکل ۴-۳۴-الف] مشابه است، با این تفاوت که بایت دوّم و سوّم این فریم یک عدد ۱۶ بیتی را تشکیل داده و محتوای آن مشخص می‌کند که برای مبادلهٔ تعداد مشخص بایت ، به چه اندازه پهنای باند نیاز است. فریم تقاضای پهنای باند دارای بخش حمل داده (Payload) و کد کشف خطا برای کلّ فریم نیست.

اگر چه می‌توان مفصلاً در خصوص 802.16 شرح داد ولی در اینجا به بحث خاتمه می‌دهیم. برای آگاهی بیشتر مستقیماً به استاندارد آن مراجعه کنید.

۶-٤ بلوتوث (Bluetooth)

در سال ۱۹۹۸ شرکت ال.ام. اریکسون علاقمند شد تا گوشی تلفنهای همراه تولیدی او بتوانند بصورت بی‌سیم به ابزارهای دیگر (مثل PDA) وصل شوند. اریکسون و چهار شرکت دیگر (آی‌بی‌ام. اینتل. نوکیا و توشیبا) یک «گروه SIG»[1] تشکیل دادند تا استانداردی بی‌سیم برای اتصال ابزارهای مخابراتی /رایانه‌ای و ابزارهای جانبی آنها طراحی کنند که بردی کوتاه، مصرف توان پائین و قیمتی ارزان داشته باشد. نام این پروژه، بلوتوث انتخاب شد که برگرفته از نام «هرالد بلاتاند دوّم» (مشهور به Bluetooth) ، یکی از پادشاهان وایکینگ است (۹۸۱_۹۴۰) که دانمارک و نروژ را با هم متحد کرد (البته با زور و بدون کابل!!!)

اگر چه تفکر اصلی، رهایی از شرّ کابلهای مابین دستگاه‌های دیجیتالی بود ولی به سرعت در حوزه‌های دیگر نیز گسترش یافت و به تدریج در حیطهٔ شبکه‌های محلی بی‌سیم نیز وارد شد. اگرچه گسترش و رشد این استاندارد، روز بروز کاربرد آنرا بیشتر می‌کرد ولی در عوض چالشهائی بین این استاندارد و 802.11 پدید آورد. نقطه شدّت این چالش آنجاست که این دو سیستم از لحاظ الکتریکی با یکدیگر تداخل فرکانسی دارند. اشاره به این نکته مهم است که شرکت هیولت پاکارد چندین سال قبل از آن شبکه‌ای مبتنی بر نور مادون‌قرمز برای وصل بی‌سیم دستگاه‌های جانبی کامپیوتر عرضه کرده بود، ولی استقبال چندانی از آن نشد.

فارغ از همهٔ اینها، در ژولای ۱۹۹۹ گروه طراح بلوتوث، مشخصات هزار و پانصد صفحه‌ای از نسخهٔ ۱ آن یعنی V1.0 را منتشر نمود. به فاصلهٔ کوتاهی، گروه استانداردسازی IEEE که در اندیشه تدوین استاندارد 802.15 برای «شبکه‌های شخصی بی‌سیم» بودند مستندات استاندارد بلوتوث را به عنوان مبنای کار خود برگزیدند و شروع به پالایش و تکمیل آن نمودند. اگر چه استانداردسازی چیزی که مشخصات تفصیلی و مشروح آن در اختیار است و پیاده‌سازیهای متنوع و ناسازگار ندارد (که نیاز به یکنواخت‌سازی و هماهنگی داشته باشد) عجیب به نظر می‌رسد

---

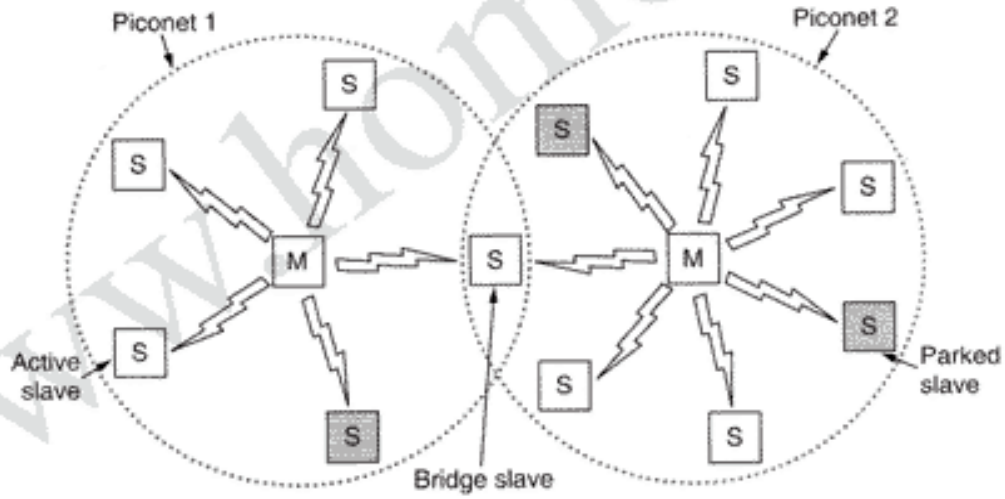
۱. SIG به معنای گروهی با گرایش خاص یا همان کنسرسیوم است.

ولی تاریخ نشان داده که وجود یک «استاندارد باز» که توسط سازمانی بی‌طرف مثل IEEE تدوین و مدیریت می‌شود عموماً کاربری یک تکنولوژی را ترویج و ترغیب خواهدکرد. اگر بخواهیم اندکی دقیقتر سخن بگوئیم باید اشاره کنیم که توصیف استاندارد بلوتوث برای سیستمی کامل تدوین شده که از لایهٔ فیزیکی تا لایهٔ کاربرد را در بر می‌گیرد درحالیکه کمیته IEEE 802.15 فقط لایه‌های فیزیکی و پیوند داده را استانداردسازی کرده و باقیماندهٔ پشتهٔ پروتکلی خارج از برنامه این استاندارد است.

هرچند IEEE اولین «**استاندارد شبکهٔ شخصی**»[1] (PAN) را در سال ۲۰۰۲ با عنوان 802.15.1 به تصویب رساند ولی هنوز کنسرسیوم بلوتوث فعال و سرگرم بهبود و توسعهٔ آنست. اگر چه نسخهٔ استاندارد عرضه شده توسط کنسرسیوم بلوتوث و IEEE یکی نیستند ولی انتظار می‌رود بزودی به یک استاندارد واحد همگرا شوند.

٤ـ۶ـ۱ معماری بلوتوث

اجازه بدهید بررسی سیستم بلوتوث را با مروری سریع بر دستاوردها و اهداف آن آغاز نمائیم. واحد پایه در سیستم بلوتوث یک «**پیکونت**» (Piconet) است که از یک «گره اصلی» (Master Node) و حداکثر هفت «گره پیرو فعّال» (Active Slave Node) به فاصلهٔ حداکثر ده‌متر ، تشکیل شده است. در یک فضای بزرگ و واحد می‌توان چندین پیکونت داشت و حتی می‌توان آنها را از طریق یک گره که نقش پل (Bride) ایفاء می‌کند به هم متصل کرد (به شکل ۴-۳۵ نگاه کنید). به مجموعه‌ای از پیکونتهای متصل بهم اصطلاحاً Scatternet (شبکه متفرق/پراکنده) گفته می‌شود.



شکل ۴ـ۳۵. دو پیکونت می‌توانند با اتصال بهم یک Scatternet تشکیل بدهند.

در یک پیکونت علاوه بر هفت گره فعال پیرو، می‌تواند تا ۲۵۵ گره غیرفعال وجود داشته باشد. اینها دستگاه‌هائی هستند که گره اصلی آنها را در حالت استراحت و کم‌توان وارد کرده تا مصرف باطری آن کاهش یابد. یک ایستگاه در حالت غیر فعال هیچ کاری نمی‌تواند انجام بدهد به جز آنکه به سیگنال فعال‌سازی خود یا سیگنال Beacon که از گره اصلی می‌رسد، پاسخ بدهد. به غیر از این حالات، دو حالت میانی در مصرف توان به نامهای حالت Hold و Sniff نیز وجود دارد که در اینجا بدان نخواهیم پرداخت.

دلیل طراحی Master/Slave (اصلی/ پیرو) آن بود که طراحان آن در نظر داشتند قیمت کل سیستم بلوتوث پیاده‌سازی شده برروی تراشه، زیر پنج دلار باشد. نتیجهٔ این تصمیم‌گیری آنست که گره‌های پیرو [مثل صفحه

---

۱. Personal Area Network

کلیدها، موس، چاپگر] تقریباً غیر هوشمند و ساده هستند و اساساً آنچه را که گره اصلی (Master) به آنها دستور بدهد اجرا می‌کنند. یک پیکونت سیستمی مبتنی بر TDM متمرکز (Centralized TDM) است که در آن هستهٔ مرکزی (یعنی گره اصلی یا Master) بر سیگنال ساعت نظارت دارد و تعیین می‌کند که چه دستگاهی و در کدام برش زمانی (Slot) مخابره داشته باشد. تبادل اطلاعات صرفاً بین گره مرکزی و گره‌های پیرو انجام می‌شود و ارتباط مستقیم دو گره پیرو [مثلاً دو صفحه کلید یا دو چاپگر] ممکن نیست.

۲-۶-٤ کاربردهای بلوتوث

بیشتر پروتکلهای شبکه فقط کانالی را بین چند مولفهٔ مخابراتی ، سازماندهی و ایجاد می‌کنند و اجازه می‌دهند طراحان برنامه‌های کاربردی به پیاده‌سازی هر آنچه که مورد نیاز است بپردازند. به عنوان مثال 802.11 مشخص نکرده که کاربران باید صرفاً از کامپیوتر کیفی خود برای خواندن ایمیل یا جستجو در وب یا هر چیز دیگری بهره بگیرند. برعکس، در تشریح بلوتوث نسخهٔ V1.1 از ۱۳ کاربرد مختلف که باید از آنها پشتیبانی شود، نام برده شده و برای هر یک، پشتهٔ پروتکلی متفاوتی ارائه گردیده است. متاسفانه این راهکار به پیچیدگی بسیار زیاد منتهی می‌شود و ما از آن صرف نظر خواهیم کرد. این سیزده کاربرد که «**پروفایل**» نام گرفته‌اند در شکل ۴-۳۶ فهرست شده‌اند. با نگاهی اجمالی به پروفایلها ممکن است به آنچه که کنسرسیوم بلوتوث در پی انجام آن بوده بیشتر پی ببریم.

نام پروفایل	عملکرد
Generic access	پروسیجرهایی برای مدیریت لینک
Service discovery	پروتکلی برای کشف سرویسهای عرضه شده
Serial port	جایگزینی برای کابل معمولی پورت سریال
Generic object exchange	مدل ارتباطی بین سرویس‌دهنده و مشتری برای جابجایی (انتقال) اشیا را تعریف می‌کند.
LAN access	پروتکل ارتباطی بین کامپیوتر همراه و شبکه محلی ثابت (با کابل سیمی)
Dial-up networking	امکان برقراری تماس یک کامپیوتر کیفی را از طریق تلفن همراه فراهم می‌آورد.
Fax	امکان ارتباط بین یک دستگاه دورنگار بی‌سیم و تلفن همراه را فراهم می‌آورد.
Cordless telephony	ارتباط بین یک دستگاه گوشی تلفن بی‌سیم و ایستگاه ثابت و محلی آن را برقرار می‌کند.
Intercom	امکاناتی برای واکی تاکی دیجیتال
Headset	امکان ارتباط از طریق هندزفری (Handsfree) را فراهم می‌آورد.
Object push	روشی برای مبادله اشیا ساده
File transfer	عرضه کننده امکانات عمومی بیشتر جهت انتقال فایل
Synchronization	امکان سنکرون‌سازی داده‌های یک PDA با کامپیوتری دیگر را فراهم می‌آورد.

شکل ۴-۳۶. پروفایلهای بلوتوث.

«**پروفایل عمومی دسترسی**» (Generic Access) حقیقتاً یک برنامهٔ کاربردی نیست بلکه بیشتر یک زیربناست که بر اساس آن برنامه‌های کاربردی حقیقی ساخته و پیاده می‌شوند. وظیفهٔ اصلی آن ارائه تمهیداتی است که بتوان بین گره اصلی (Master) و گره‌های پیرو (Slave) یک کانال مطمئن برقرار و آنرا حفظ کرد. «**پروفایل تشخیص خدمات**» (Service Discovery) که آنهم تقریباً عمومی و کلّی است توسط دستگاه‌ها برای آگاهی از خدماتی که دیگر دستگاه‌ها ارائه می‌دهند، استفاده می‌شود. تمام دستگاه‌های مبتنی بر بلوتوث موظف به پیاده‌سازی این دو پروفایل هستند. بقیه پروفایلها اختیارند.

«**پروفایل درگاه سریال**» (Serial Port) یک پروتکل انتقال است که بقیه پروفایلها از آن بهره می‌گیرند. این

پروفایل یک درگاه سریال را شبیه‌سازی می‌کند و بطور خاص برای کاربردهای قدیمی که به خط سریال نیاز دارند، سودمند است.

«**پروفایل عمومی مبادلهٔ شیء**» (Generic Object Exchange) یک ارتباط مبتنی بر مدل مشتری/سرویس‌دهنده ، برای انتقال داده‌ها تعریف کرده است. اگرچه همیشه مشتری، آغازکنندهٔ عملیات است ولیکن یک گره پیرو می‌تواند هم سرویس‌دهنده و هم مشتری باشد. همانند پروفایل «درگاه سریال» این پروفایل نیز زیربنای دیگر پروفایلهاست.

گروه سه‌تائی پروفایلهای بعدی به منظور کاربردهای شبکه‌ای (Networking) تعریف شده‌اند. «**پروفایل دسترسی به LAN**» اجازه می‌دهد که یک دستگاه مبتنی بر بلوتوث به یک شبکهٔ ثابت متصل شود. این پروفایل رقیب مستقیم 802.11 است. «**پروفایل شبکهٔ مبتنی بر شماره‌گیری**» (Dialup) انگیزهٔ اصلی کل این پروژه بوده است. این پروفایل اجازه می‌دهد که یک کامپیوتر کیفی بتواند به یک تلفن همراه که دارای مودم داخلی بی سیم است متصل شود. «**پروفایل دورنگار**» (Fax) شبیه به پروفایل شماره‌گیری است با این تفاوت که اجازه می‌دهد ماشینهای دورنگار بی‌سیم از طریق یک دستگاه تلفن همراه و بدون نیاز به سیم، اقدام به ارسال یا دریافت دورنگار کنند.

سه پروفایل بعدی در خصوص تلفن کاربرد دارند. پروفایل «**تلفن بی‌سیم**» (Cordless Telephony) راهی را برای اتصال گوشی یک تلفن بی‌سیم به ایستگاه ثابت است. در حال حاضر تلفنهای بی‌سیم خانگی را نمی‌توان به عنوان تلفن همراه به کارگرفت ولی در آینده شاید تلفن بی‌سیم و تلفن همراه در هم ادغام شوند. «**پروفایل Intercom**» این امکان را فراهم می‌کند تا دو تلفن ، شبیه به «واکی‌تاکی» (Walkie/Talkie) بهم متصل گردند. نهایتاً «**پروفایل گوشی** –Headset–» امکان ارتباط بی‌سیم بین گوشی و ایستگاه ثابت (مثل هندزفری –Hands Free-) را فراهم می‌کند که به عنوان مثال برای صحبت با تلفن در حین رانندگی مفید است.

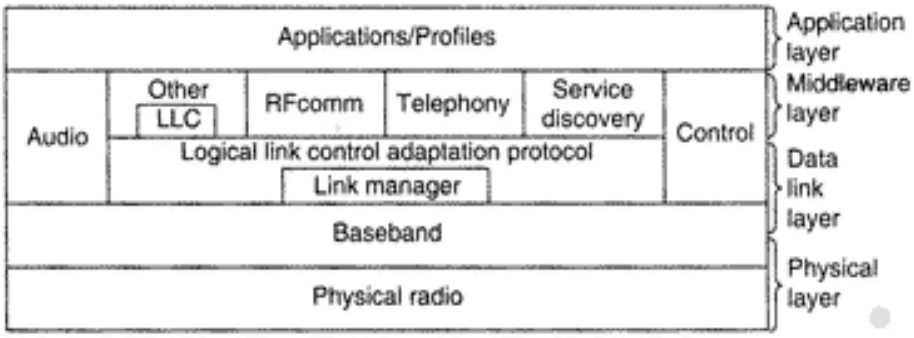
سه پروفایل باقیمانده برای مبادلهٔ اشیاء بین دو ابزار تعریف شده است. اشیاء می‌توانند فایلهای داده، تصویر یا کارتهای تجاری باشند. «**پروفایل سنکرواسیون**» ، برای بار کردن داده در درون کامپیوتر کیفی یا PDA (کامپیوترهای دستی) در حین ترک منزل و جمع آوری اطلاعات پس از برگشت، مفید است.

آیا واقعاً نیاز بوده که تمام این کاربردها به تفصیل تحلیل شوند و برای هرکدام پشته پروتکلی متفاوتی تعریف گردد؟ شاید نه! امّا بخشهای متفاوت این استاندارد را گروههای کاری مختلف طراحی کرده‌اند و چون هر یک از این گروهها برروی نیازهای خاص خود متمرکز بودند، در نتیجه هر یک پروفایل موردنظر خود را پدید آوردند. برای توجیه این موضوع به قانون کانوی (Conway's Law) بیندیشید. (در یکی از شماره‌های مجلهٔ Datamation در آوریل ۱۹۶۸ ، ملوین کانوی مشاهدات خود را بدین نحو منتشر کرد که اگر n شخص را به نوشتن یک کامپایلر بگمارید، آنچه که بدست خواهید آورد یک کامپایر n–pass است [یعنی کامپایلری که در آن تعداد مراحل ترجمهٔ یک برنامه ، n گذر می‌باشد.] بعبارت عام ساختار نهائی یک نرم‌افزار آیینهٔ تمام‌نمای ترکیب گروهی است که آن را تولید کرده‌اند. شاید می‌شد که به جای سیزده پشته پروتکلی به دو پشتهٔ کلّی بسنده کرد: یکی برای انتقال فایل و دیگری برای انتقال بی‌درنگ جریان اطلاعات.

۳-۶-٤ پشتهٔ پروتکلی بلوتوث

استاندارد بلوتوث پروتکلهای متعددی دارد که بطور ناموزون در چند لایه گروه‌بندی شده‌اند. ساختار لایه‌ها از مدل OSI ، مدل TCP/IP ، مدل 802 یا هر مدل شناخته شدهٔ دیگر تبعیت نمی‌کند. با این وجود IEEE در حال اصلاح بلوتوث است تا با مدل 802 سازگارتر شود. معماری پروتکل بلوتوث که توسط کمیته 802 اصلاح شده ، در شکل ۴-۳۷ مشاهده می‌شود.

لایهٔ زیرین، «**لایهٔ رادیو فیزیکی**» است که تقریباً متناظر با لایهٔ فیزیکی از مدل OSI یا مدل 802 می‌باشد. این



شکل ۴ـ۳۷. نسخهٔ 802.15 از معماری پروتکل بلوتوث.

لایهٔ با انتقال رادیوئی و مدولاسیون سرکار دارد. بسیاری از ملاحظاتی که در طراحی این لایه باید مورد توجه قرار می‌گرفت آن بود که سیستم ارزان قیمت باشد و بطور انبوه در بازار عرضه شود.

«لایهٔ باند پایه» (Baseband) از جهاتی شبیه به زیرلایهٔ MAC است ولیکن مولفه‌هائی از لایهٔ فیزیکی را نیز در بر می‌گیرد. این لایه با مسائلی مثل چگونگی نظارت گره اصلی (Master) بر برشهای زمانی و چگونگی گروه‌بندی این برشهای زمانی در قالب فریمها ، سرکار دارد.

سپس لایه‌ای شامل یک گروه از پروتکلهای مرتبط با هم، تعریف شده است. مدیر لینک (Link Manager) عملیات ایجاد کانالهای منطقی بین دستگاه‌ها، شامل «مدیریت توان مصرفی»، «احراز هویت» (Authentication) و «کیفیت خدمات» (QoS) را بر عهده دارد. «پروتکل تطبیق کنترل لینک منطقی» (که اغلب L2CAP گفته می‌شود) وظیفه دارد لایه‌های بالایی را از درگیری با جزئیات ارسال ، راحت کند. این لایهٔ مشابه با استاندارد زیرلایهٔ LLC 802 است ولی از لحاظ مسائل فنّی با آن متفاوت است. دو پروتکل «کنترل» و «صدا» همانگونه که از نامشان بر می‌آید با مسائل انتقال صدا و عملیات کنترل سروکار دارند. برنامه‌های کاربردی می‌توانند بدون نیاز به L2CAP ، مستقیماً این دو پروتکل را به خدمت بگیرند.

لایهٔ بعدی یک لایهٔ میانی است (Middleware) و تلفیقی از پروتکلهای متفاوت را در برمی گیرد. در این لایه از پروتکل IEEE 802 LLC ، بمنظور سازگاری با دیگر شبکه‌های سری 802 استفاده شده است. پروتکلهای RFcomm ، Telephony و Service Discovery صرفاً مرتبط با بلوتوث هستند. RFcomm (Radio Frequency Communication) ، پروتکلی جهت شبیه‌سازی استاندارد درگاه سریال (Serial port) است که در تمام PCها از آن برای اتصال صفحه کلید، موس، مودم و امثال آن استفاده می‌شود. این پروتکل برای آن طراحی شده تا بتوان از دستگاه‌های قدیمی بسهولت استفاده کرد. «پروتکل تلفنی» (Telephony) پروتکلی بی‌درنگ است که برای سه پروفایل مبتنی بر انتقال صدا بکار می‌آید. این پروتکل همچنین تنظیم و قطع ارتباط را بر عهده دارد. نهایتاً پروتکل «تشخیص خدمات» (Service Discovery) برای کشف و تشخیص انواع خدماتی که درون شبکه عرضه می‌شود، کاربرد دارد.

بالاترین لایه، محّل قرار گرفتن انواع برنامه‌های کاربردی و پروفایلها است. این لایه برای انجام کار از خدمات پروتکلهای موجود در لایه‌های زیر بهره می‌گیرد. هر برنامهٔ کاربردی، زیرمجموعه‌ای از پروتکلهای مختص به خود را به خدمت می‌گیرد. ابزارهای ویژه‌ای مثل گوشی بی‌سیم (Headset) بسته به نوع برنامهٔ کاربردی آنها، فقط به برخی از پروتکل‌ها نیازمندند.

در بخشهای بعدی سه لایهٔ پائینی از پشتهٔ پروتکلی بلوتوث را بررسی خواهیم کرد چرا که تقریباً متناظر با زیرلایه‌های فیزیکی و MAC است.

٤ـ٦ـ٤ لایهٔ رادیوئی در بلوتوث

لایهٔ رادیوئی بیتها را از گره اصلی به گره پیرو و بالعکس، منتقل می‌کند. این لایه ، سیستمی با توان کم و برد ده متر است که در باند فرکانسی 2.4GHz ISM عمل می‌کند. این باند به ۷۹ کانال یک مگاهرتزی تقسیم می‌شود. مدولاسیون به کاررفته FSK (Frequency Shift Keying) و هر هرتز (هر سیکل) معادل یک بیت است که جمعاً نرخ یک مگابیت بر ثانیه را در اختیار می‌گذارد ولی بیشتر این پهنای باند به دلیل سربار تلف می‌شود. برای تخصیص مناسب این کانالها از روش پرش فرکانس در طیف گسترده (Spread Spectrum) با نرخ پرش 1600 hops/sec و Dwell time معادل ۶۲۵ میکروثانیه بهره گرفته شده است.

چون بلوتوث و 802.11 ، هر دو در باند 2.4GHz ISM و دقیقاً در همان ۷۹ کانال کار می‌کنند لذا با یکدیگر تداخل فرکانسی خواهند داشت. از آنجایی که پرش فرکانس در بلوتوث سریعتر از 802.11 است لذا دستگاه‌های مبتنی بر بلوتوث به احتمال بیشتری در انتقال 802.11 اخلال خواهد کرد تا 802.11 در بلوتوث! چون 802.11 و 802.15 هر دو استانداردهای IEEE هستند لذا IEEE به دنبال راه حلّی برای این مشکل می‌گردد ولی حَل این مشکل چندان ساده نیست چرا که هر دو سیستم، بدلیل مشابهی از این باند فرکانسی بهره گرفته‌اند: زیرا، برای استفاده از این باند فرکانسی، به اخذ هیچ مجوزی نیاز نیست. استاندارد 802.11a از باند دیگر ISM (باند 5GHz) استفاده می‌کند ولیکن برد کمتری نسبت به 802.11b دارد (بدلیل ماهیت فیزیکی امواج رادیوئی این باند) لذا استفاده از 802.11a راه حل مناسبی نخواهد بود. برخی از شرکتها این مشکل را با ممنوعیت استفاده از بلوتوث حل کرده‌اند. راه حل بازاری این مشکل آنست که صبر کنیم تا عاقبت شبکه‌ای که از لحاظ اقتصادی و سیاسی جایگاه مستحکم‌تری در بازار پیدا کرد، از طرف مقابل بخواهد تا استاندارد خود را برای حَل مشکل تداخل، اصلاح نماید. در این خصوص مطالبی در مرجع (Lansford et al. ,2001) ارائه شده است.

٤ـ٦ـ٥ لایهٔ باند پایه در بلوتوث

لایهٔ باند پایه شبیه‌ترین بخش بلوتوث با زیرلایهٔ MAC است. این لایه، دنبالهٔ بیتهای خام را به فریمها تبدیل می‌کند و بدین منظور چندین قالب مهم فریم تعریف نموده است. در ساده‌ترین حالت، گره اصلی در هر **پیکونت** دنباله‌ای از برشهای زمانی ۶۲۵ میکروثانیه‌ای (Time Slot) تولید می‌کند، با این توصیف که ارسال داده‌های گره اصلی در برشهای زمانی با شمارهٔ زوج انجام می‌شود و گره‌های پیرو (Slaves) در برشهای زمانی فرد ارسال می‌نمایند. این روش مشابه با روش تسهیم زمانی (TDM) معمولی است که در آن ، گره اصلی نیمی از برشهای زمانی را در اختیار دارد و بقیه گره‌ها (حداکثر هفت گره) در نیم دیگر سهیم هستند. ارسال هر فریم می‌تواند ۱ ، ۳ یا ۵ برش زمانی طول بکشد.

در هر پرش فرکانسی ۲۵۰ تا ۲۶۰ میکروثانیه طول خواهد کشید تا مدار رادیوئی بتواند پایدار شود. پایداری سریعتر نیز ممکن است ولی هزینهٔ پیاده‌سازی بیشتری دارد. برای فریمهای که فقط به یک برش زمانی نیاز دارند پس از هر پرش فرکانسی، ۳۶۶ بیت از کل ۶۲۵ بیت باقی خواهد ماند. از این مقدار ۱۲۶ بیت به «**کد دسترسی**» (Access Code) و «**سرآیند**» اختصاص دارد و ۲۴۰ بیت ، برای داده‌ها باقی می‌ماند. وقتی پنج برش زمانی به هم ملحق می‌شوند [برای ارسال فریمهائی به طول ۵ برش] تنها به یک زمان پایداری نیاز خواهد بود و طبعاً زمان کوتاهتری برای زمان پایداری، تلف می‌شود و ۵×۶۲۵=۳۱۲۵ بیت در پنج برش زمانی ارسال می‌گردد که از این مقدار ۲۷۸۱ بیت برای ارسال داده در اختیار «**لایهٔ باند پایه**» خواهد بود. بنابراین در بلوتوث فریمهای طولانی کارآمدتر از فریمهای کوچک هستند.

هر فریم برروی یک کانال منطقی که اصطلاحاً «لینک بین گره اصلی و گره پیرو» نام دارد ، ارسال خواهد شد. دو نوع لینک وجود دارد: لینک اول ACL است (Asynchronous Connection-Less) که برای ارسال داده‌ها در

برشهای زمانی نامنظم کاربرد دارد. این داده‌ها از لایهٔ L2CAP در سمت فرستنده تولید و در سمت گیرنده تحویل لایهٔ L2CAP می‌شوند. ترافیک داده‌های ACL مبتنی بر روش «**بیشترین تلاش**» (Best Effort) ارسال می‌شود ولی هیچ تضمینی در تحویل آنها نیست. فریمها می‌توانند گم شده یا از بین بروند، بدون آنکه ارسال مجدد شوند. هر گره پیرو فقط می‌تواند یک لینک ACL با گره اصلی داشته باشد.

لینک دیگر، SCO (Synchronous Connection Oriented) نام دارد که برای ارسال داده‌های بی‌درنگ، مثل ارتباط تلفنی کاربرد دارد. این نوع کانال با تخصیص برشهای زمانی مشخص در هر دو جهت، ایجاد می‌شود. بدلیل آنکه لینکهای SCO نسبت به زمان حساس هستند فلذا فریمهای ارسالی برروی این لینک هرگز ارسال مجددً (Retransmit) نخواهند شد، در عوض از روش «تصحیح مستقیم خطا» استفاده شده تا اطمینان بیشتری داشته باشد.[ارسال مجدد فریمهای خراب بر عهدهٔ لایهٔ بعدی است و در این لایه –در صورت امکان– به تصحیح خطا بسنده می‌شود. –م] هر گره پیرو می‌تواند حداکثر سه لینک SCO با گره اصلی داشته باشد. هر لینک SCO می‌تواند یک کانال صدا مبتنی بر PCM با نرخ 64000 بیت برثانیه را حمل کند.

٤ـ۶ـ۶ لایهٔ L2CAP در بلوتوث

لایهٔ L2CAP سه دسته عملیات مهم را بر عهده دارد: اول آنکه بسته‌هائی با طول حداکثر ۶۴کیلوبایت را از لایه‌های بالائی پذیرفته و آنها را جهت انتقال ، به فریمهای کوچکتری می‌شکند. در سمت مقابل این فریمها مجدداً به بستهٔ اصلی بازسازی خواهند شد.

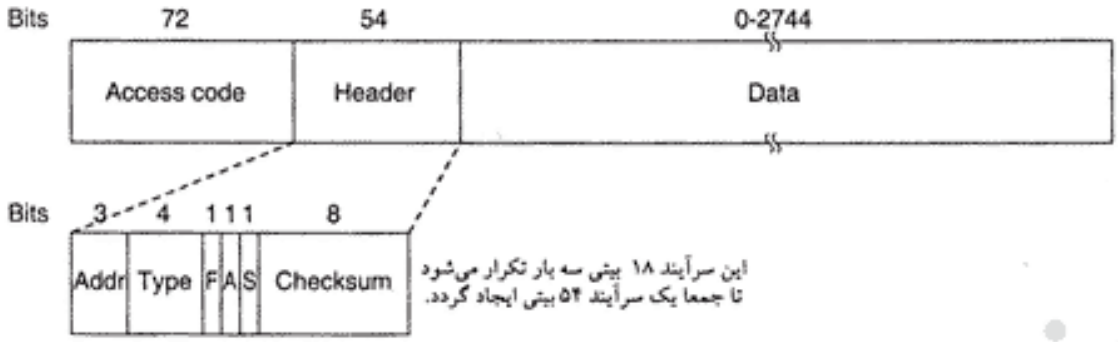
دوّم آنکه این لایهٔ عمل جمع‌آوری و توزیع بسته‌هائی که از چندین مبداء آمده [یا به چندین مقصد می‌روند] را برعهده دارد. وقتی یک بسته بازسازی می‌شود، لایهٔ L2CAP تعیین خواهد کرد که باید به کدام پروتکل در لایهٔ بالاتر (مثلاً RFcomm یا Telephony) تحویل شود.

سوم آنکه این لایه، عملیات تامین «کیفیت خدمات» (QoS) را بر عهده دارد، چه در هنگام ایجاد لینک و چه در خلال عملکرد طبیعی. همچنین در زمان ایجاد لینک، بر سر اندازهٔ حداکثر و مُجاز طول داده، توافق صورت می‌گیرد تا ابزارهائی با طول بسته بزرگ از ارسال چنین بسته‌ای به ابزارهائی با طول بستهٔ کوچک اجتناب کنند. از آنجایی که تمام ابزارهای مبتنی بر بلوتوث نمی‌توانند بسته‌هائی با طول ۶۴کیلوبایت را بپذیرند فلذا به این ویژگی نیاز است.

٤ـ۶ـ۷ ساختار فریم در بلوتوث

چندین نوع قالب فریم در بلوتوث وجود دارد که مهمترین آنها در شکل ۴–۳۸ نشان داده شده است. این فریم با فیلد «کد دسترسی» (Access Code) شروع می‌شود که عموماً هویت یک گره اصلی (Master) را مشخص خواهد کرد تا بدینگونه یک گره پیرو که در برد رادیوئی دو گره اصلی قرار دارد، گیرندهٔ حقیقی ترافیک داده‌ها را مشخص نماید. سپس یک سرآیند ۵۴ بیتی آمده که شامل فیلدهای معمولی زیرلایهٔ MAC است. سپس فیلد داده قرار گرفته که حداکثر ۲۷۴۴ بیت را (برای انتقال در پنج برش زمانی) در بر می‌گیرد. در فریمهائی که تنها در یک برش زمانی ارسال می‌شوند، قالب فریم همین است با این تفاوت که فیلد دادهٔ آنها حداکثر ۲۴۰ بیت است.

حال اجازه بدهید به فیلدهای سرآیند، نگاهی سریع بیندازیم. «فیلد آدرس» هویت گیرندهٔ فریم را از بین هشت دستگاه فعال در هر پیکونت مشخص می‌کند. «فیلد Type»، اولاً نوع فریم را (از بین انواع ACL ، SCO ، POLL یا NULL)، ثانیاً روش تصحیح خطای داده‌ها را و ثالثاً تعداد برشهای زمان که ارسال فریم جاری بدان نیاز دارد را مشخص می‌نماید. «بیت Flow» توسط گره‌های پیرو و زمانی تنظیم [فعال] می‌شود که بافر آنها پر شده باشد و نتوانند دادهٔ بیشتری دریافت کنند. این بیت، شکل ابتدائی کنترل جریان داده‌ها، به حساب می‌آید. «بیت



شکل ۴-۳۸. قالب کلّی فریم داده در بلوتوث.

Acknowledgement» بدین منظور است تا دریافت صحیح یک فریم از طرف مقابل، در فریم ارسالی جاسازی و اعلام شود [یعنی فرآیند Piggybacking]. «بیت Sequence» برای شماره‌گذاری فریمهاست تا بسته‌های تکراری کشف شوند. در بلوتوث پروتکل ارسال مجدّد، روش «توقف و انتظار» (Stop & wait) است و طبعاً یک بیت برای شماره‌گذاری فریمها کفایت می‌کند. در ادامه «فیلد هشت بیتی Checksum» برای کشف خطای احتمالی در سرآیند تعریف شده است. کلّ این سرآیند ۱۸ بیتی سه بار تکرار می‌شود تا سرآیند ۵۴ بیتی نشان داده شده در شکل ۴-۳۸بوجود آید. در سمت گیرنده، با یک مدار ساده سه نسخهٔ تکراری هر بیت بررسی می‌شود. اگر هر سه بیت مثل هم بودند، آن بیت پذیرفته می‌شود در غیر اینصورت، بیتی که بیشترین تکرار را دارد قبول می‌شود. بدین ترتیب، برای ارسال یک سرآیند ۱۰ بیتی ظرفیتی معادل ۵۴ بیت صرف می‌شود. دلیل آن این بوده که برای ارسال مطمئن داده‌ها در محیطی سرشار از نویز و با استفاده از ابزاری ارزان قیمت و توانی ناچیز (2.5mW) و قدرت پردازش پائین، به افزونگی بسیار زیادی نیاز خواهد بود.

برای فیلد داده در فریمهای ACL، قالبهای متفاوتی تعریف شده است. فریمهای SCO ساده‌تر هستند: فیلد داده همیشه ۲۴۰ بیتی است. سه گزینهٔ دیگر نیز تعریف شده که در آنها مقدار واقعی داده‌ها ۸۰، ۱۶۰ یا ۲۴۰ بیتی است و باقیماندهٔ بیتها برای تصحیح خطا به کار می‌آیند. در مطمئن‌ترین نسخه (یعنی دادهٔ ۸۰بیتی)، بخش داده همانند سرآیند، سه بار متوالی تکرار می‌شود.

از آنجایی که گرو پیرو (Slave) تنها می‌تواند از برشهای زمانی با شمارهٔ فرد استفاده نماید فلذا در هر ثانیه ۸۰۰ برش زمانی بدست خواهد آورد. (همینطور گره اصلی) بدین ترتیب با فیلد دادهٔ ۸۰ بیتی، ظرفیت کانال از سمت گره پیرو به سمت گره اصلی معادل ۶۴۰۰۰ بیت بر ثانیه خواهد بود (ظرفیت کانال از گره اصلی به گره پیرو نیز ۶۴۰۰۰ بیت است) لذا این کانال دقیقاً برای یک کانال صوتی دو طرفهٔ مبتنی بر PCM کافی است. (دلیل انتخاب نرخ 1600 Hops/sec برای تغییر فرکانس همین بوده است). این اعداد و ارقام بدین معنا هستند که یک کانال صوتی دو طرفهٔ PCM با نرخ ۶۴۰۰۰ بیت برثانیه، در حالت مطمئن [یعنی وقتی داده‌های ۸۰بیتی با سه بار تکرار ارسال می‌شوند]، کّل پهنای باند موجود در پیکونت را (علیرغم پهنای باند 1 Mbps آن)، اشباع خواهد کرد. با گزینهٔ نامطمئنتر (یعنی ۲۴۰ بیت در هر برش زمانی بدون هیچگونه افزونگی یا تکرار) می‌توان از حداکثر سه کانال صوتی همزمان حمایت کرد و به همین دلیل حداکثر سه لینک SCO برای هر گره پیرو مجاز شمرده شده است.)

مطالب فراوانتری در خصوص بلوتوث می‌توان گفت که از حوصله این کتاب خارج است. برای آگاهی بیشتر به مراجع زیر مراجعه نمائید.

Bhagwat, 2001; Bisdikian, 2001; Bray and Sturman, 2002; Haartsen, 2000; Johansson et al., 2001; Miller and Bisdikian, 2001; Sairam et al., 2002)

۷-٤ هدایت در سطح لایهٔ پیوند داده‌ها (Data Link Layer Switching)

بسیاری از سازمانها دارای LANهای متعددی هستند و تمایل دارند آنها را به هم متصل کنند. شبکه‌های محلی (LAN) را می‌توان از طریق دستگاه‌هائی که در لایهٔ پیوند داده‌ها عمل می‌کنند و «پل» (Bridge) نامیده می‌شوند به هم متصل نمود. «پلها» برای مسیریابی و هدایت داده‌ها، آدرسهای «لایهٔ پیوند داده‌ها» را بررسی می‌نمایند. از آنجایی که قرار نیست محتوای فیلد داده (از فریمهائی که باید هدایت شوند) بررسی گردد لذا این فریمها می‌توانند بسته‌های IPv4 (که اکنون در اینترنت به کار می‌رود)، IPv6 (که در آینده در اینترنت به کار گرفته خواهد شد)، بسته‌های AppleTalk ، ATM ، OSI یا هر نوع بستهٔ دیگر را در خود حمل کنند. برخلاف پل ، «مسیریابها» آدرس درون بسته‌ها را بررسی کرده و بر این اساس، آنها را هدایت (مسیریابی) می‌کنند. اگرچه این تعریف تمایز بین «مسیریاب» و «پل» را تعیین می‌کند ولی پیشرفتهای جدیدی مثل ابداع شبکهٔ LAN مبتنی بر سوئیچ (Switched Ethernet) ، آب را گل کرده و به نحوی که در بخشهای آتی بدانها خواهیم پرداخت این تمایز را ابهام‌آلود کردند! در بخشهای بعدی به پلها و سوئیچها و بالاخص آنهائی که برای اتصال شبکه‌های محلی 802 به کار می‌آیند، نگاهی خواهیم انداخت. برای بررسی دقیق پلها، سوئیچها و عناوین مهم در این خصوص، به مراجع (Perlman , 2000) مراجعه نمائید.

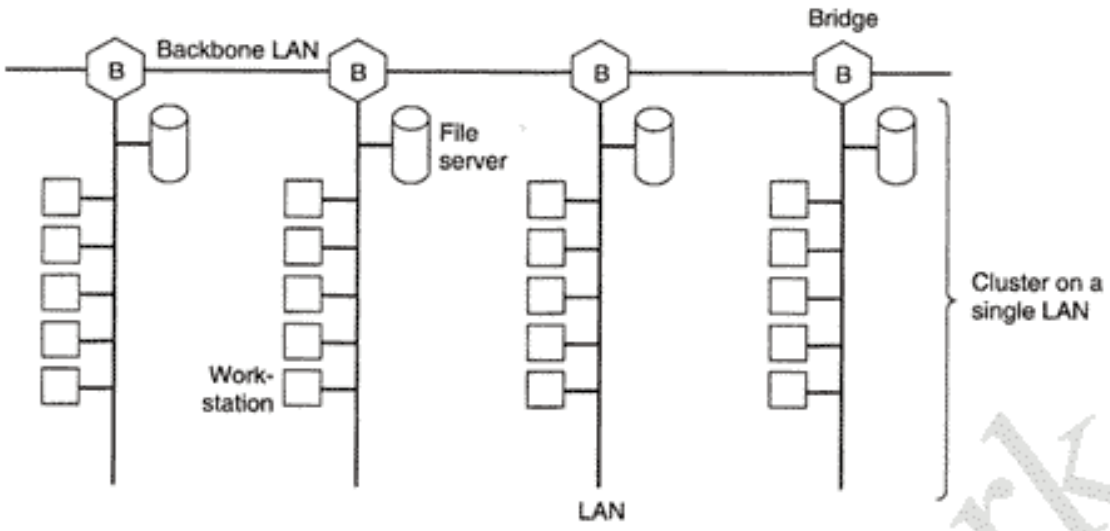
قبل از پرداختن به تکنولوژی پل ، بررسی شرایطی که در آن ، استفاده از پلها سودمند است، خالی از لطف نخواهد بود. شش دلیل ارائه می‌کنیم که چرا یک سازمان واحد، ممکن است دارای چندین LAN باشد.

اوّل آنکه بسیاری از دانشگاه‌ها و بخشهای مختلف شرکتها ، LAN مختص به خود را دارند تا بتوانند کامپیوترهای شخصی ، ایستگاه‌های کاری (Workstation) و سرویس‌دهنده‌های خاص خود را به هم متصل کنند. از آنجایی که بخشهای مختلف یک موسسه، اهداف متفاوتی را دنبال می‌کنند لذا در هر بخش، فارغ از آنکه دیگر بخشها چه می‌کنند، LAN متفاوتی پیاده می‌شود. دیر یا زود نیاز می‌شود که این LANها با یکدیگر تعامل و ارتباط داشته باشند. دراین مثال، پیدایش LANهای متعدد ناشی از اختیار و آزادی مالکان آن بوده است.

دوّم آنکه ممکن است سازمانها به صورت جغرافیائی در ساختمانهائی با فاصله قابل توجه، پراکنده باشند. شاید داشتن چندین LAN مجزا در هر ساختمان و وصل آنها از طریق «پلها» و لینک‌های لیزری ارزان‌تر از کشیدن یک کابل واحد بین تمام سایتها تمام شود.

سوم آنکه گاهی برای تنظیم بار و تعدیل ترافیک ، لازم است که یک LAN منطقی و واحد به چندین LAN کوچکتر تقسیم شود. به عنوان مثال در بسیاری از دانشگاه‌ها هزاران ایستگاه کاری در اختیار دانشجویان و هیئت علمی قرار گرفته است. عموماً فایلها در ماشینهای سرویس‌دهندهٔ فایل نگهداری می‌شوند و حسب تقاضای کاربران برروی ماشینشان منتقل و بارگذاری می‌شود. مقیاس بسیار بزرگ این سیستم مانع از آن می‌شود که بتوان تمام ایستگاه‌های کاری را در یک شبکهٔ محلی واحد قرار داد چرا که پهنای باند مورد نیاز بسیار بالا خواهد بود. در عوض، مشابه با شکل ۴-۳۹ از چندین LAN که توسط «پل» به هم متصل شده، استفاده می‌شود. هر شبکهٔ LAN گروهی از ایستگاه‌ها و سرویس دهندهٔ فایل خاص خود را دربرمی‌گیرد که بدین ترتیب، بیشتر ترافیک در حوزهٔ یک LAN واحد محدود می‌شود و بار زیادی به ستون فقرات شبکه اضافه نخواهد شد.

اشاره به این نکته ارزشمند است که اگرچه عموماً شبکه‌های LAN را به صورت یک کابل چنداتصالی (Muitidrop) با ساختار باس ترسیم می‌کنیم (نمایش کلاسیک) ولیکن امروزه اغلب آنها توسط هاب و خصوصاً سوئیچها پیاده‌سازی می‌شوند. با این حال یک کابل طولانی با چندین ماشین متصل به آن، با یک هاب که ماشینها را از درون ، بهم متصل می‌کند، از لحاظ عملکرد یکسان هستند. در هر دو حالت تمام ماشینها به یک «حوزهٔ تصادم» (Collision Domain) یکسان متعلقند و تمام آنها برای ارسال فریم از پروتکل CSMA/CD استفاده می‌کنند.



شکل ۴ـ۳۹. چندین LAN از طریق یک ستون‌فقرات بهم متصل شده‌اند تا ظرفیت کلّ حمل بار آن از ظرفیت یک LAN واحد بیشتر شود.

شبکه‌های مبتنی بر سوئیچ متفاوت هستند و اگر چه قبلاً آنها را بررسی کرده‌ایم ولی باز هم نگاهی به آنها خواهیم انداخت .

چهارم آنکه در برخی از شرایط اگرچه یک شبکهٔ محلی واحد از نظر حجم بار کفایت می‌کنند ولیکن فاصلهٔ فیزیکی بین ماشینهای دور، بسیار زیاد است. (مثلاً بیش از ۲/۵ کیلومتر در اترنت). حتی اگر عملیات کابل‌کشی ساده باشد ولیکن شبکه، در اثر تاخیر بسیار زیاد رفت و برگشت سیگنال (Round Trip Delay) کار نخواهد کرد. تنها راه حَل آنست که LAN به چند بخش تقسیم شده و بین آنها پل نصب گردد. با استفاده از پل ، می‌توان فاصلهٔ فیزیکی کل شبکه را افزایش داد.

پنجم مسئله قابلیت‌اعتماد است؛ برروی یک LAN واحد، یک گره خراب که دنباله‌ای پیوسته از خروجی آشغال تولید می‌کند قادر است کَل شبکه را فلج نماید. پلها را می‌توان در نقاط حساس قرار داد تا یک گره خراب و مغشوش نتواند کَل سیستم را مختل کند. برخلاف یک تکرارکننده (Repeater) که ورودی خود را بی قید و شرط بازتولید می‌نماید یک پل را می‌توان به نحوی برنامه‌ریزی کرد تا در خصوص آنچه که هدایت می‌کند یا هدایت نمی‌کند تصمیم آگاهانه بگیرد.

ششم و آخر آنکه پلها می‌توانند به امنیت اطلاعات در یک سازمان کمک نمایند. بیشتر کارتهای واسط شبکه‌های LAN دارای حالتی به نام «حالت بی‌قید» (Promiscuous mode) هستند که در چنین حالتی تمام فریمهای جاری برروی شبکه تحویل گرفته می‌شود، نه فریمهائی که دقیقاً به آدرس او ارسال شده‌اند. جاسوسان و فضولان به این ویژگی علاقمند هستند. با قرار دادن پلها در نقاط مختلف و اطمینان از عدم هدایت اطلاعات حساس به بخش‌های نامطمئن، مسئول سیستم می‌تواند بخشهائی از شبکه را از دیگر بخشها جدا کرده تا ترافیک آنها به خارج راه پیدا نکرده و در اختیار افراد نامطمئن قرار نگیرد.

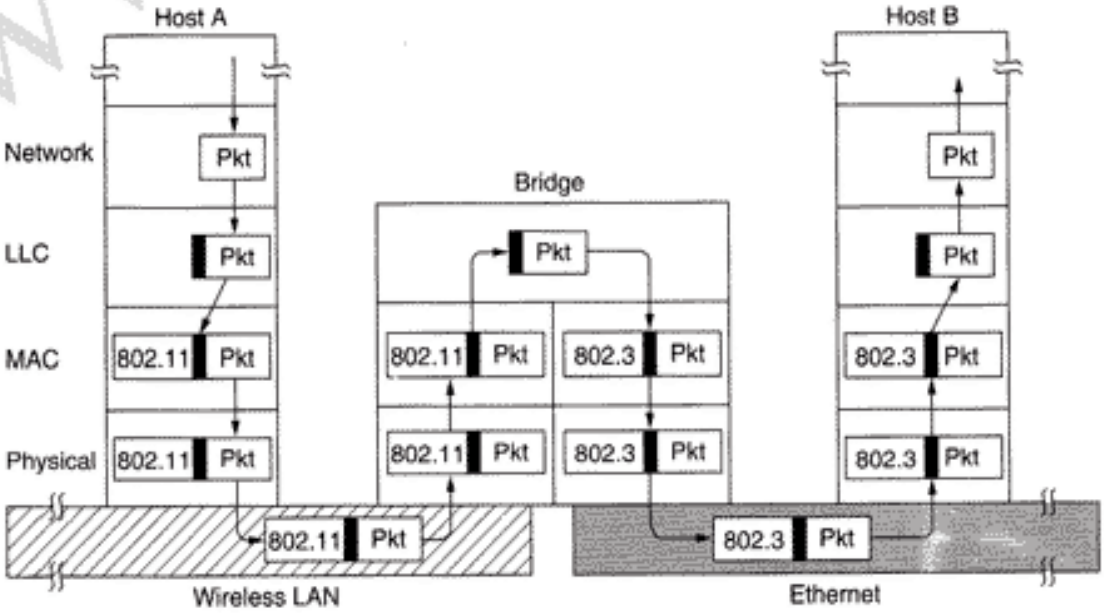
هدف آرمانی آنست که پلها کاملاً نامرئی (شفاف–transparent) باشند، بدین معنا که بتوان ماشینی را از یک بخش از شبکه به بخش دیگر منتقل کرد بدون آنکه به هیچگونه تغییری در سخت‌افزار، نرم‌افزار یا جداول پیکربندی نیاز باشد. همچنین باید این امکان وجود داشته باشد که تمام ماشینهای یک بخش از شبکه بتوانند فارغ از آنکه نوع LAN آنها چیست با ماشینهای بخش دیگر، مبادلهٔ اطلاعات داشته باشند. این هدف گاهی برآورده می‌شود ولیکن نه همیشه!

٤-٧-١ پلهائی از 802.x به 802.y

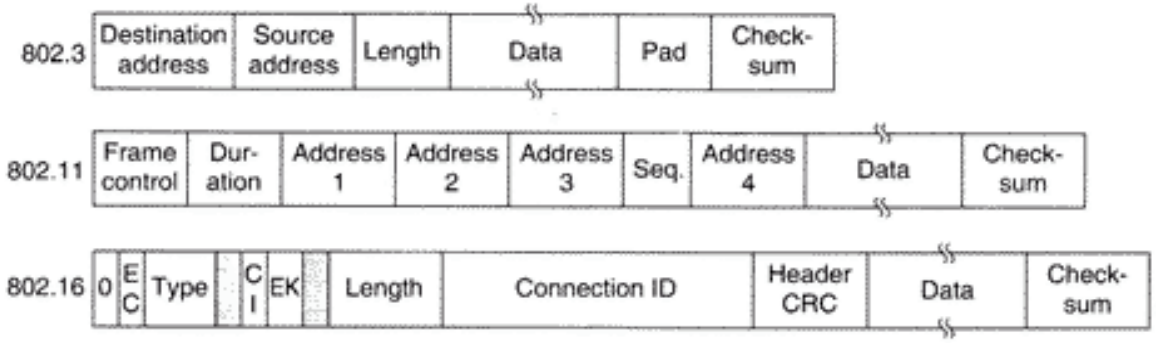
پس از بررسی آنکه چرا به پلها نیاز است اجازه بدهید بدین سوال بپردازیم که عملکرد آنها چگونه است؟ شکل ۴-۴۰ عملکرد یک پل ساده با دو درگاه (Port) را به تصویر کشیده است. ماشین میزبان A در یک شبکهٔ محلی بی‌سیم، بسته‌ای برای ارسال به ماشین میزبان و ثابت B در شبکهٔ اترنت (802.3) (که از طریق پل به شبکهٔ بی‌سیم متصل شده) آماده کرده است. این بسته از زیرلایهٔ LLC عبور کرده و سرآیند LLC به آن اضافه می‌شود (این سرآیند در شکل به صورت سیاه نشان داده شده است). سپس به زیرلایهٔ MAC تحویل شده و در آنجا نیز سرآیند 802.11 به‌ابتدای آن افزوده می‌شود. (یک بخش انتهایی نیز به آخر فریم اضافه خواهد شد که در شکل نشان داده نشده است.) این واحد داده ، در هوا منتشر و توسط ایستگاه ثابت دریافت می‌شود؛ پس از بررسی ، ایستگاه ثابت متوجه می‌گردد که باید آنرا به سمت اترنت ثابت هدایت کند. پس از رسیدن آن به پل (که شبکهٔ 802.11 را به شبکهٔ 802.3 متصل کرده)، پل کار دریافت آن از لایهٔ فیزیکی را شروع کرده و آنرا به سمت زیرلایه‌های بالا هدایت می‌کند. در زیرلایهٔ MAC از پل، سرآیند 802.11 حذف می‌شود. بستهٔ اصلی (به همراه سرآیند LLC) تحویل زیرلایهٔ LLC از پل می‌شود. در این مثال، بسته به سمت شبکه محلی 802.3 رهسپار است، لذا بسته از طریق بخش 802.3 در پل به سمت پائین حرکت کرده و نهایتاً برروی شبکهٔ اترنت منتقل می‌شود. دقت کنید که یک پل که k شبکهٔ مختلف را به هم متصل می‌کند دارای k زیرلایهٔ MAC و تعداد k لایهٔ فیزیکی است (به ازای هر نوع یکی).

تا اینجا به نظر می‌رسد که انتقال فریم از یک LAN به LAN دیگر ساده است اما واقعیت این چنین نیست. در این بخش، برخی از دشواریهای ساخت یک پل را که انواع مختلف شبکه‌های LAN (و همچنین شبکهٔ MAN) سری 802 را به هم متصل می‌کند، متذکر می‌شویم. ما برروی شبکه‌های 802.3، 802.11 و 802.16 متمرکز خواهیم شد ولی انواع دیگر آن هم وجود دارد که هر کدام مسائل خاص خود را دارند.

برای شروع باید یادآوری کرد که هر LAN قالب فریم خاص خود را دارد. (شکل ۴-۴۱ را ببینید) اگرچه تفاوتهایی که بین قالب فریم در شبکه‌های اترنت ، «توکن رینگ» (Token Ring) و «توکن باس» (Token Bus) وجود دارد بیشتر ناشی از اتکاء به نفس شرکتهای بزرگ ابداع‌کننده آنها (یعنی به ترتیب زیراکس، آی‌بی‌ام و جنرال‌موتورز) و همچنین شرایط زمانی آن دوران بوده ولیکن تفاوت شبکه‌های کنونی تقریباً لازم و



شکل ۴-۴۰. عملکرد یک پل از شبکهٔ 802.11 به 802.3.



شکل ۴ـ۴۱. انواع قالب فریمهای IEEE 802 (طول هر فریم در شکل، مقیاس اندازهٔ واقعی آن نیست.)

واقعی است. به عنوان مثال فیلد Duration (طول زمان) که در 802.11 وجود دارد ناشی از ماهیت پروتکل MACAW و همچنین عدم توانایی شنود کانال در این شبکه می‌باشد (برخلاف اترنت). در نتیجه، انتقال یک فریم بین دو شبکهٔ متفاوت LAN ، مستلزم دگرگونی در قالب فریمهاست که طبعاً نیاز به زمان CPU ، محاسبهٔ مجدّد کدهای جدید کشف خطا (Checksum) دارد و این احتمال نیز پدید می‌آید که در اثر خرابی بیتها در حافظه پل، خطاهای غیرقابل کشف، داده‌ها را آلوده کند.

مشکل دوم آنست که LANهای به هم متصل‌شده الزاماً با سرعت مشابهی کار نمی‌کنند. وقتی «پل» یک دنبالهٔ پی‌درپی از فریمها را از LAN سریع برای هدایت به LAN کندتر می‌پذیرد قادر نخواهد بود با همان سرعتی که فریم را دریافت می‌کند از دست آنها رهائی یابد. به عنوان مثال هرگاه یک شبکهٔ اترنت گیگابیتی با بالاترین سرعت ، بیتها را به سوی شبکهٔ یازده مگابیت‌درثانیه‌ای 802.11b LAN روانه کند، پل باید بتواند آنها را موقتاً در حافظه نگهداری کند و این حافظه نباید سرریز شود. [که با چنین سرعتی بعید نیست.] در ضمن، برخی پلها سه یا چندین شبکهٔ LAN را به هم متصل می‌کنند. در چنین پلهایی اگر چندین شبکه بطور همزمان تلاش کنند فریمهای خود را به یک LAN مشابه بفرستند مشکل سرریز شدن حافظه بوجود خواهد آمد، حتی اگر تمام LANها با سرعت یکسانی کار کنند.

مشکل سوّم که خطیرترین مشکل بالقوهٔ پلها محسوب می‌شود آنست که در شبکه‌های محلی و مختلف سری 802 ، حداکثر طول فریم ، متفاوت است. این مشکل زمانی بروز می‌کند که یک فریم بزرگ باید به سوی شبکه‌ای هدایت شود که آن LAN قادر به دریافت آن نیست. شکستن فریم به تعدادی قطعه، خارج از حیطهٔ وظائف این لایه است. در تمام پروتکلهای این لایه فرض برآنست که فریمها یا می‌رسند یا نمی‌رسند و هیچ تمهیدی برای شکستن یک فریم به قطعات و بازسازی آن اندیشیده نشده است. نمی‌توان گفت که چنین پروتکلی ابداع نشده است. چنین پروتکلی ابداع شده و وجود هم دارد ولیکن در حیطه وظایف پروتکلهای پیونده داده نیست. پلها نیز نباید به محتوای هر فریم کاری داشته باشند. اساساً (در چنین وضعیتی) راه حلی برای این مشکل وجود ندارد. فریمهای بسیار طولانی باید به جای انتقال، حذف شوند. [جهت اجتناب از هرگونه مداخله در پیکربندی سخت‌افزار یا نرم‌افزار]

نکته بعدی امنیت داده‌هاست. شبکه‌های 802.11 و 802.16، هر دو از رمزنگاری در سطح لایهٔ پیوند داده‌ها پشتیبانی می‌کنند در حالیکه اترنت چنین امکانی ندارد. بدین معنا که خدمات متنوع رمزنگاری عرضه شده در شبکه‌های بی‌سیم، در خلال ورود ترافیک به شبکهٔ اترنت از دست می‌رود. بدتر از آن، اینکه اگر یک دستگاه در شبکهٔ بی‌سیم از رمزنگاری در سطح لایهٔ پیوند داده‌ها بهره گرفته باشد هیچ راهی برای رمزگشائی آن هنگام

دریافت در یک ایستگاه اترنت وجود ندارد. از طرفی اگر ایستگاه بی‌سیم از رمزنگاری استفاده نکند ترافیک داده‌ها از طریق لینک هوائی در معرض شنود همگان قرار می‌گیرد.

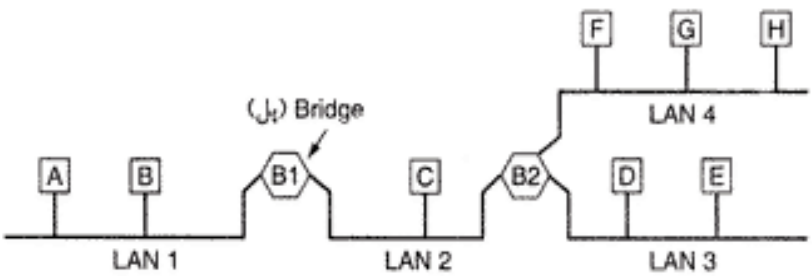
یک راه‌حل برای مشکل امنیت آنست که رمزنگاری در لایه‌های بالائی انجام شود ولیکن در این روش ایستگاه 802.11 باید بداند که آیا با ایستگاه دیگری در شبکهٔ 802.11 صحبت می‌کند (تا از رمزنگاری در لایهٔ پیوند داده بهره بگیرد) یا نه (تا از چنین امکانی استفاده نکند). وادار کردن ایستگاه به تصمیم‌گیری «**شفافیت**» را از بین خواهد برد.

نکتهٔ آخر مسئله «کیفیت خدمات» است. هر دو شبکهٔ 802.11 و 802.16 این خدمات را به نحو متفاوتی در اختیار می‌گذارند: اوّلی در «حالت PCF» و دوّمی با استفاده از «اتصال با نرخ ارسال ثابت» (Constant Bit Rate Connection). در شبکهٔ اترنت چیزی به نام کیفیت خدمات بی‌مفهوم است و بدین ترتیب ترافیک هر یک از این دو شبکه در حین عبور از اترنت «کیفیت خدمات» را از دست می‌دهند. [به عنوان مثال اگر چه می‌توان میزان حداکثر تاخیر در دو شبکهٔ اوّل را تضمین کرد ولی در اترنت چنین تضمینی وجود ندارد و با وصل این دو شبکه نمی‌توان تاخیر کلّ را تضمین نمود و مقدار آن تصادفی خواهد بود. -م]

۲-۷-٤ بهم‌بندی شبکه‌ها به صورت محلی (Local Internetworking)

در بخش قبلی به مشکلاتی که در وصل دو شبکهٔ محلّی نوع IEEE 802 بروز می‌کند، پرداختیم. با این وجود در سازمانهای بزرگ با شبکه‌های LAN متعدد، متصل کردن تمام آنها به یکدیگر مشکلات گوناگونی را بوجود می‌آورد، حتی اگر تمام آنها از نوع اترنت باشند. حالت آرمانی آنست که بتوان به راحتی از سازمان بیرون رفت و چند پل مبتنی بر استاندارد IEEE خریداری کرد، سپس با وصل تمام کابلها به پلها ، شبکه فوراً و به درستی بکار بیفتد. نباید به هیچ تغییر سخت‌افزاری، نرم‌افزاری، تنظیم سوئیچها، بارگذاری یا تغییر در جداول مسیریابی یا پارامترهای آن یا هر تغییر دیگری نیاز باشد. فقط باید کابلها را وصل کرد و رفت!! به علاوه عملکرد طبیعی هیچکدام از شبکه‌های LAN نباید تحت‌الشعاع قرار بگیرد. به عبارت دیگر، پلها بایستی کاملاً «شفاف» باشند (یعنی از دیدگاه سخت‌افزار و نرم‌افزار غیرقابل رویت باشند). شگفت آنکه این کار عملاً ممکن است. حال اجمالاً بررسی کنیم که این کار جادوئی به چه نحو انجام می‌شود!

در ساده‌ترین حالت، یک «پل شفاف» در «حالت بی‌قید» (Promiscuous Mode) عمل کرده و تمام فریمهای جاری برروی تمام LANهائی را که بدانها متصل است، می‌پذیرد. به عنوان یک مثال، به پیکربندی شکل ۴-۴۲ توجه نمائید. پل B1 به شبکه‌های محلی ۱ و ۲ متصل شده و پل B2 نیز به شبکه‌های محلی ۲ و ۳ و ۴ متصل است. فریمی که به پل B1 می‌رسد ولی مقصد آن ماشین A است می‌تواند فوراً توسط پل نادیده انگاشته شود. (حذف شود) زیرا این فریم برروی LAN صحیحی [که به مقصد ختم می‌شود] قرار گرفته است ولی فریمی که از LAN 1 به مقصد C یا F دریافت می‌شود باید منتقل شود.



شکل ۴-۴۲. یک پیکربندی از شبکه‌های متصل‌بهم با چهار شبکهٔ محلی و دو پل.

وقتی فریمی دریافت می‌شود، پل در ابتدا باید تصمیم بگیرد که آیا باید آنرا حذف کند یا باید آنرا منتقل نماید؛ سپس در صورت نیاز به انتقال، باید مشخص شود که به کدام LAN هدایت گردد. این تصمیم‌گیری با جستجوی آدرس مقصد درون یک جدول بزرگ در حافظهٔ پل انجام می‌گیرد. (به این جدول ، Hash Table گفته می‌شود.) این جدول فهرست تمام ماشینهای مقصد را در اختیار دارد و می‌تواند تعیین کند که این ماشینها به کدامیک از خطوط پل تعلق دارند. مثلاً جدول پل B2 می‌تواند مشخص کند که ماشین A به LAN 2 تعلق دارد، زیرا تمام آنچه که لازم است B2 بداند آنست که فریمهائی با مقصد A را برروی چه شبکه‌ای ارسال کند. در واقع برای پل B2 مهم نیست که این فریم، بعداً چگونه هدایت و منتقل می‌شود.

وقتی پلها برای اولین بار به کار می‌افتند تمام جداول Hash خالی هستند. هیچیک از پلها نمی‌دانند که هر یک از ماشینهای مقصد در کجا قرار گرفته‌اند، لذا برای انتقال فریم از «**الگوریتم سیل‌آسا**» (Flooding Algorithm) استفاده می‌نماید یعنی تمام فریمهای ورودی که مقصدشان ناشناخته است برروی تمام شبکه‌هائی که پل بدانها متصل است، ارسال می‌شود (البته به استثنای شبکه‌ای که فریم از آن دریافت شده است). به مرور زمان، پل متوجه خواهد شد که هر ماشین مقصد، در کجا قرار گرفته است. (طبق الگوریتمی که در زیر بدان اشاره خواهیم کرد.) هرگاه یک ماشین مقصد شناسائی شد، فریمهائی که بعداً بدان مقصد روانه می‌شوند توسط پل برروی LAN مناسب هدایت خواهد شد و از روش سیل‌آسا استفاده نمی‌شود.

الگوریتمی که توسط پلهای شفاف به کار گرفته می‌شود روش «**یادگیری غیرمستقیم**» (Backward Learning) است. قبلاً اشاره شد که پل در «حالت بی‌قید» کار می‌کند لذا هر فریمی را که برروی یکی از شبکه‌های متصل به او مبادله می‌شود، می‌بیند (و همچنین دریافت و پردازش می‌نماید). با نگاهی به آدرس مبداء هر فریم، پل می‌تواند بفهمد که کدام ماشین از طریق کدام LAN قابل دسترسی است. به عنوان مثال در شکل ۴-۴۲ فرض کنید که پل B1 فریمی را بر روی LAN 2 می‌بیند که توسط C تولید شده است؛ بدین ترتیب متوجه می‌شود که می‌توان از طریق LAN 2 به C رسید و بدین ترتیب در جدول Hash خود درج می‌کند که فریمهای روانه به سمت C باید از طریق LAN 2 منتقل و هدایت شوند. بعداً تمام فریمهائی که به مقصد C و از طریق LAN 1 به پل وارد می‌شوند منتقل می‌شوند ولی فریمهائی که به مقصد C ولی از طریق LAN 2 به پل می‌رسند حذف خواهند شد.

با خاموش یا روشن شدن پلها و ماشینها و یا جابجائی آنها در سطح شبکه، توپولوژی شبکه تغییر می‌کند. برای آنکه توپولوژی شبکه به صورت پویا دنبال شود وقتی یک «**درایه**» (Entry) در جدول هر پل درج می‌گردد زمان دریافت فریم نیز در آن «درایه» یادداشت می‌شود. هرگاه بعداً فریمی که آدرس مبداء آن قبلاً در جدول درج گردیده، دریافت شود زمان درج شده در درایهٔ متناظر آن، با زمان فعلی بهنگام‌سازی می‌شود. بدین ترتیب زمان درج شده در هر «درایه» (Entry) آخرین زمانی که فریمی از آن ماشین دریافت شده را تعیین می‌کند.

بطور متناوب، یک پروسه در درون پل، جدول Hash را جستجو و پویش می‌کند و تمام درایه‌هائی را که برای بیش از چند دقیقه به هنگام نشده‌اند، حذف می‌نماید. بدین ترتیب اگر یک کامپیوتر از شبکهٔ خود جدا و در ساختمان جابجا شده یا به شبکهٔ دیگری متصل شود پس از گذشت چند دقیقه عملیات عادّی خود را از سرمی‌گیرد و نیازی به مداخله و تنظیمات دستی نیست. البته این الگوریتم بدین معنا هم هست که اگر کامپیوتری برای چندین دقیقه ساکت باشد (یعنی هیچ داده‌ای نفرستد) درایهٔ متناظر با او از جدول پلها حذف شده و از آن به بعد هر ترافیکی که برایش ارسال می‌شود بروش سیل‌آسا هدایت خواهد شد مگر آنکه بعداً خودش فریمی ارسال کند [و آدرس او در جدول پلها درج شود. -م]

روند مسیریابی فریمهای ورودی پل ، به شبکهٔ LAN مبداء (Source LAN) و شبکهٔ LAN مقصد (Destination LAN) بستگی دارد. این روال به ترتیب ذیل است:

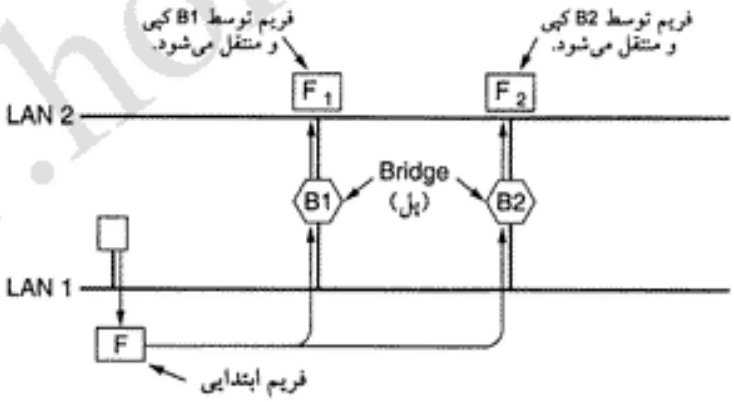
۱. اگر شبکهٔ LAN مبدأ و مقصد یکسان هستند فریم را نادیده بگیر.
۲. اگر شبکه‌های مبدأ و مقصد متفاوت هستند فریم را منتقل کن.
۳. اگر شبکهٔ مقصد ناشناخته است بروش «سیل‌آسا» عمل کن.

این الگوریتم به ازای ورود هر فریم باید یکبار اعمال گردد. یک تراشهٔ خاص VLSI عملیات جستجو و به هنگام‌سازی جدول و درایه‌های (Entries) را در عرض چند میکروثانیه انجام می‌دهد.

۴-۷-۳ پلهای مبتنی بر درخت پوشا (Spanning Tree)

برخی از سایتها برای افزایش قابلیت اعتماد، بین دو LAN دو یا چند پل موازی نصب می‌کنند.(به شکل ۴-۴۳ نگاه کنید.) ولیکن چنین آرایشی مشکلات دیگری را ایجاد خواهد کرد چراکه در ساختار توپولوژی، حلقه ایجاد می‌شود.

چنین مشکلاتی را می‌توان بکمک مثال شکل ۴-۴۳ و با مشاهدهٔ روند هدایت فریمی از ماشین F به مقصدی ناشناخته، تحلیل کرد. هر پل طبق قاعدهٔ عمومی در مواجهه با فریمهائی که به مقصد ناشناخته روانه هستند از روش «سیل‌آسا» استفاده می‌کند یعنی در اینجا فریم برروی LAN 2 منتقل می‌شود. [فریم منتقل شده بر روی LAN 2 را F_2 بنامید.] اندکی بعد، پل ۱ فریم F_2 را می‌بیند که مقصد آن ناشناخته است و آنرا برروی LAN 1 هدایت می‌کند و فریم F_3 تولید می‌شود. (فریم F_3 در شکل نشان داده نشده است.) بروش مشابه پل ۲ فریم F_1 را برروی LAN 1 منتقل کرده و F_4 تولید می‌شود. (F_4 نیز نشان داده نشده است) [F_1 ، F_2 ، F_3 ، F_4 نسخه‌های مشابه با فریم F هستند که توسط پل منتقل می‌شوند.] حال پل ۲ فریم F_4 را منتقل و پل ۲ فریم F_3 را منتقل می‌نماید و این دور باطل تا ابد ادامه دارد.



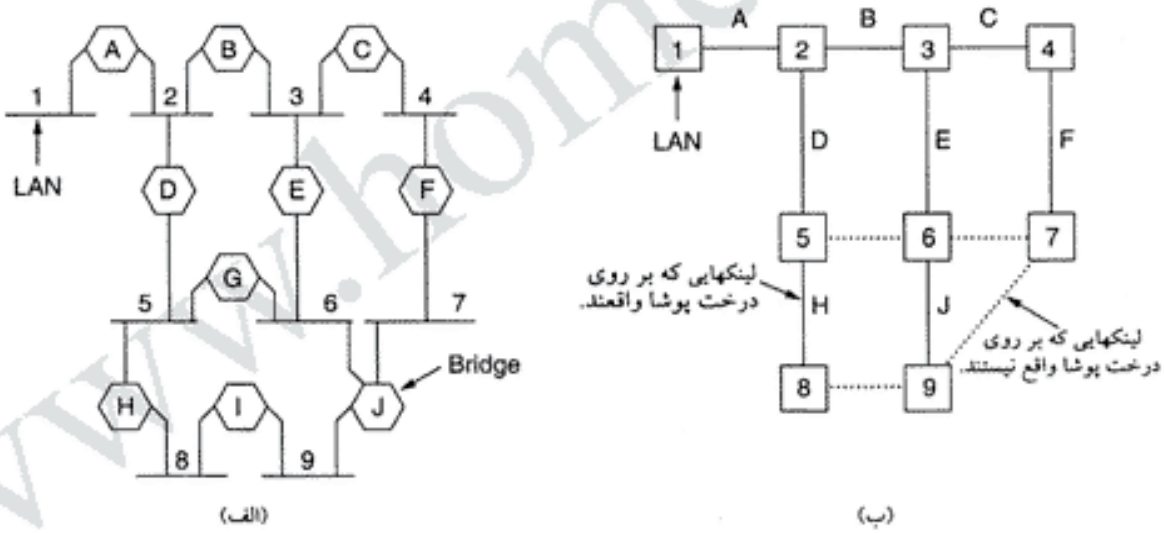
شکل ۴-۴۳. دو پل شفاف (نامرئی) موازی.

راه حلّ این مشکل آنست که پلها با یکدیگر ارتباط و محاوره داشته باشند و توپولوژی واقعی را به صورت یک «**درخت پوشا**» (Spanning Tree) که در آن به تمام LANها مسیر دسترسی وجود دارد، در نظر بگیرند. در نتیجه برخی از اتصالات بالقوه که بین LANها وجود دارد، نادیده گرفته می‌شود تا مجازاً یک توپولوژی بدون حلقه ایجاد شود. به عنوان مثال در شکل ۴-۴۴-الف ، نُه شبکهٔ LAN را می‌بینیم که توسط ده پل متصل شده‌اند. این پیکربندی را می‌توان در قالب گرافی در نظر گرفت که در آن، هر کدام از LANها یک «گره» تلقی می‌شوند. هر «کمان» (Arc) اتصال دو LAN توسط پل را نشان می‌دهد. با حذف برخی از «کمانها» (که در شکل ۴-۴۴-ب به صورت خط چین نشان داده شده) ، این گراف به یک درخت پوشا کاهش می‌یابد. با استفاده از این درخت، بین دو LAN دقیقاً یک مسیر وجود دارد. پس از آنکه پلها درخت پوشا را تشکیل دادند، هدایت تمام فریمها (بین

شبکه‌های محلّی) از ساختار درخت پوشا تبعیت خواهد کرد. از آنجایی که بین هر مبداء و مقصد در شبکه فقط و فقط یک مسیر وجود دارد لذا ایجاد حلقه غیرممکن است.

برای ایجاد درخت پوشا، پلها بایستی یک پل را به عنوان ریشهٔ این درخت انتخاب کنند. جهت این انتخاب، پلها شمارهٔ سریال خود را که توسط کارخانهٔ سازنده تنظیم و یکتا بودن آن در کلّ دنیا تضمین شده است، به صورت فراگیر (Broadcast) به اطلاع همه می‌رسانند. پلی که دارای کوچکترین شمارهٔ سریال است به عنوان «ریشه» انتخاب می‌شود. سپس درختی با کوتاهترین مسیر که از ریشه شروع شده و تمام پلها و LANها را در بر می‌گیرد، ایجاد می‌شود. این درخت همان «درخت پوشا» است. اگر یک پل یا شبکهٔ LAN از کار بیفتد، این درخت از نو محاسبه می‌شود.

نتیجهٔ این الگوریتم آنست که بین هر LAN و ریشه و بدین ترتیب از ریشه به بقیهٔ LANها، یک مسیر یکتا ایجاد می‌شود. اگر چه این درخت تمام LANها را در بر می‌گیرد ولیکن تمام پلها لزوماً در این درخت قرار نمی‌گیرند (برای اجتناب از حلقه). حتی پس از ایجاد درخت پوشا و در خلال عملکرد طبیعی، این الگوریتم به صورت متناوب اجرا می‌شود تا هر گونه تغییر در توپولوژی به صورت خودکار تشخیص داده شده و درخت اصلاح شود. این الگوریتم توزیع‌شده که برای ایجاد درختهای پوشا مورد استفاده قرار می‌گیرد توسط خانم رایدیا پرلمن ابداع و به تفصیل در مرجع (Perlman,2000) تشریح شده است. همچنین این الگوریتم در IEEE 802.1D استانداردسازی شده است.

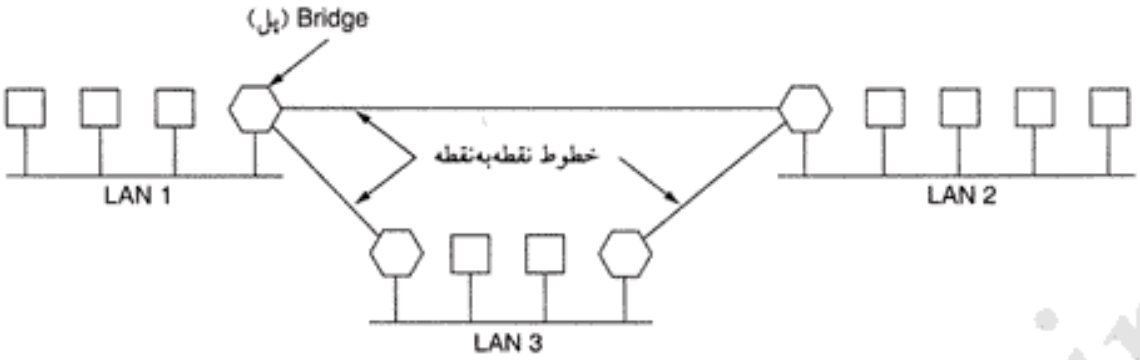


شکل ۴ـ۴۴. (الف) چند شبکهٔ LAN بهم متصل (ب) یک درخت پوشا که تمام شبکه‌های LAN را در بر می‌گیرد. (خطوط نقطه‌چین جزو درخت پوشا نیستند.)

٤ـ٧ـ٤ پلهای راه دور (Remote Bridges)

کاربرد متداول پلها آنست که دو یا چند LAN راه دور را بهم متصل کنند. بعنوان مثال ممکن است یک شرکت دارای کارخانه‌هایی در چند شهر باشد و هر کارخانه، LAN مختص به خود را داشته باشد. در حالت آرمانی باید تمام LANها به هم متصل شده باشند تا کلّ سیستم نقش یک LAN عظیم را ایفاء کند.

این هدف با قرار دادن یک پل بین هر دو LAN و وصل کردن پلها به وسیلهٔ خطوط نقطه‌به‌نقطه (مثلاً بکمک خطوط اجاره‌ای عرضه شده توسط شرکتهای تلفن) برآورده خواهد شد. در شکل ۴-۴۵، یک سیستم ساده با سه LAN به تصویر کشیده شده است. در اینجا روشهای رایج مسیریابی اعمال می‌شود. ساده‌ترین راه برای تحلیل این



شکل ۴ـ۴۵. برای اتصال شبکه‌های محلی راه دور می‌توان از پلهای راه دور بهره گرفت.

ساختار آنست که سه خط نقطه‌به‌نقطه را به مثابه یک شبکهٔ LAN بدون هیچ ماشین میزبان (Hostless LAN) ، تصور کنید. در این صورت یک سیستم معمولی با شش LAN داریم که با چهار پل به هم متصل شده‌اند. در هیچ کجا از مطالبی که تاکنون مطالعه کرده‌ایم گفته نشده که حتماً باید به یک LAN ماشین میزبان متصل شده باشد!

برای خطوط نقطه‌به‌نقطه می‌توان از پروتکلهای متنوعی بهره گرفت. یک انتخاب آنست که از یکی از استانداردهای موجود برای خطوط نقطه‌به‌نقطه مثل PPP استفاده شود و کلّ فریمهای MAC [شامل سرآیند و فیلدهای پایانی] درون فیلد حمل دادهٔ آن (یعنی Payload) قرار بگیرد. این استراتژی بشرط آنکه تمام LANها مثل هم باشند، به نحو احسن کار می‌کند و تنها مسئلهٔ باقیمانده آنست که فریمها به LAN صحیح تحویل شوند. انتخاب دیگر آنست که سرآیند و پی‌آیند[1] هر فریم MAC در پل مبداء حذف شود و باقیمانده در فیلد حمل داده از فریم مربوط به پروتکل نقطه‌به‌نقطه قرار بگیرد. سپس در پل مقصد سرآیند و پی‌آیند MAC جدیدی برای آن قطعهٔ داده تولید شود. اشکال این روش آنست که کد کشف خطایی که فریم در حین دریافت در ماشین مقصد دارد همانی نیست که توسط ماشین مبداء تولید شده و بدین ترتیب ممکن است خطاهایی که در حافظهٔ پل ، داده‌ها را آلوده می‌کند کشف نشود.

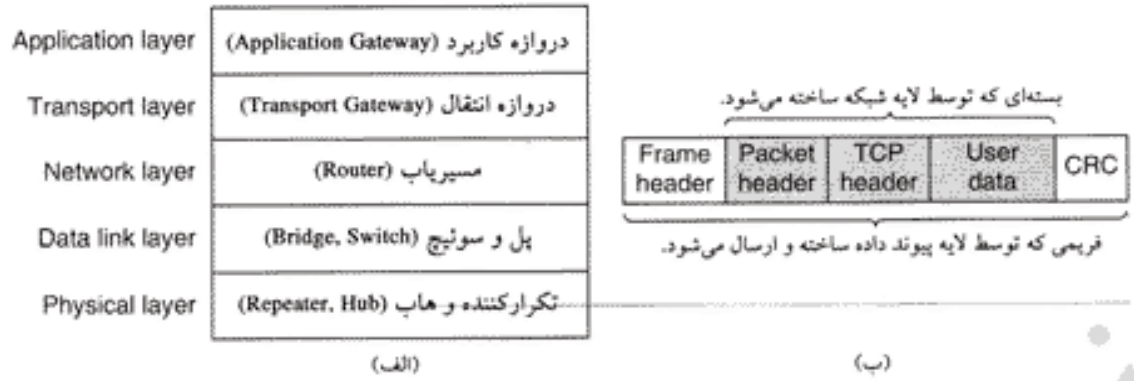
۵ـ۷ـ۴ تکرارکننده[2]، هاب[3]، پل[4]، سوئیچ[5]، مسیریاب[6] و دروازه[7]

تا اینجای کتاب روشهای گوناگونی را مرور کردیم که وظیفهٔ همگی تحویل فریمها یا بسته‌ها از یک بخش کابل به بخش دیگر است. همچنین اشاره‌ای به تکرارکننده‌ها، پلها، سوئچها، هابها، مسیریابها و دروازه‌ها داشتیم. از تمام این ابزارها به یک منظور استفاده می‌شود ولیکن تفاوتهای مشهود و نامشهود زیادی دارند. از آنجایی که این ابزارها بسیار متنوعند، لذا مروری بر همهٔ آنها و بررسی شباهتها و تفاوتهای آنها ارزشمند خواهد بود.

برای شروع باید گفت که این ابزارها (به نحوی که در شکل ۴-۴۶-الف دیده می‌شود) در لایه‌های متفاوتی عمل می‌کنند. بسته به آنکه هر یک از این ابزارها در چه لایه‌ای عمل می‌کنند مکانیزم هدایت اطلاعات متفاوت است. در قالب یک نمایشنامهٔ عمومی: کاربر مقداری اطلاعات برای ارسال به یک ماشین راه دور تولید می‌نماید. این داده‌ها تحویل لایهٔ انتقال (Transport Layer) شده و بدان سرآیند لازم اضافه می‌گردد (مثلاً سرآیند TCP). سپس، واحد اطلاعاتی حاصل به سمت لایهٔ پائین یعنی لایهٔ شبکه عبور داده می‌شود. لایهٔ شبکه سرآیند خاصّ خود را بدان افزوده و یک بستهٔ مخصوص لایهٔ شبکه ساخته می‌شود (مثلاً یک بستهٔ IP). در شکل ۴-۴۶-ب یک بستهٔ IP را می‌بینیم که به صورت خاکستری نشان داده شده است. این بسته تحویل لایهٔ پیوند داده (Data Link)

---

۱. Header and Trailer ۲. Repeater ۳. Hub ۴. Bridge
۵. Switch ۶. Router ۷. Gateway

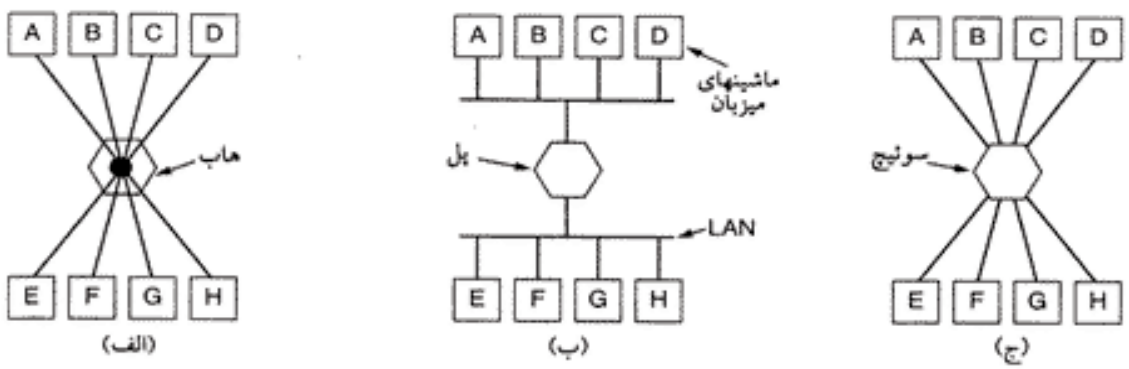


شکل ۴ـ۴۶. (الف) جایگاه هر ابزار در پشتهٔ پروتکلی (ب) فریمها، بسته‌ها و سرآیندها.

می‌شود و آن هم سرآیند خاص خود و کد کشف خطا (CRC) را بدان افزوده و فریم حاصل را جهت ارسال به لایهٔ فیزیکی تسلیم می‌کند. (مثلاً برای ارسال برروی LAN)

حال اجازه بدهید نگاهی به ابزارهای هدایت اطلاعات بیندازیم و ببینیم که این ابزارها چه ارتباطی با بسته‌ها و فریمها دارند. در پائین‌ترین سطح یعنی در لایهٔ فیزیکی به «تکرارکننده‌ها» بر می‌خوریم. تکرارکننده ابزاری است آنالوگ ، که دو قطعه کابل را بهم متصل می‌کند. سیگنالی که برروی یکی از این قطعات ظاهر گردد، تقویت (بازتولید) شده و بر روی قطعه دیگر قرار داده می‌شود. تکرارکننده‌ها هیچگونه درکی از «فریم» ، «بسته» یا «سرآیند» ندارند. آنها صرفاً با مفهوم «وُلت» آشنا هستند! به عنوان مثال در اترنت کلاسیک اجازه داده شده برای افزایش طول حداکثر کابل از ۵۰۰ متر به ۲۵۰۰ متر از چهار تکرارکننده استفاده شود.

سپس به هاب (Hub) می‌رسیم: یک هاب معمولی (از نوع غیرفعال)، دارای تعدادی خط ورودی است که این خطوط از لحاظ الکتریکی در درون هاب به هم متصل شده‌اند. فریمی که از یک خط ورودی دریافت می‌شود بر روی خطوط دیگر ارسال خواهد شد. هرگاه دو فریم بطور همزمان به هاب ارسال شوند، تصادم رخ خواهد داد؛ دقیقاً همانند اتفاقی که برروی کابل کواکسیال می‌افتد. بعبارت دیگر، کلّ هاب یک «حوزهٔ تصادم» (Collision Domain) واحد را تشکیل خواهد داد. تمام خطوط ورودی هاب، باید با سرعت یکسانی کار کنند. هابها متفاوت از تکرارکننده‌ها هستند، از آن جهت که هاب (معمولاً) سیگنالهای ورودی را تقویت نمی‌کند و طراحی آنها به گونه‌ای است که چندین کارت واسط خط (Line Card) دارند و هر یک از کارتها خود چندین ورودی دارند ولیکن در مجموع این تفاوتها ناچیز است. همانند تکرارکننده‌ها، هابها نیز آدرسهای 802 [آدرسهای MAC] را بررسی نکرده و به هیچ وجه از آنها استفاده نمی‌کنند. در شکل ۴-۴۷-الف تصویری نمادین از یک هاب نشان داده شده است.



شکل ۴ـ۴۷. (الف) یک هاب (ب) یک پل (ج) یک سوئیچ.

حال اجازه بدهید به سمت لایهٔ پیوند داده یعنی لایه‌ای که در آن «پلها» و «سوئیچها» تعریف شده‌اند حرکت کنیم. قبلاً «پلها» را تا حدّی مطالعه کردیم. یک پل دو یا چند شبکهٔ LAN را همانند شکل ۴-۴۷-ب به هم متصل می‌کند. وقتی فریمی دریافت می‌شود نرم‌افزار درون پل، آدرس مقصد را از سرآیند فریم استخراج و آنرا درون جدول خود جستجو می‌کند تا محلّی را که فریم باید بدانجا ارسال شود، بیابد. در اترنت، این آدرس همان فیلد ۴۸ بیتی آدرس در شکل ۴-۱۷ است. شبیه به هاب، پلهای پیشرفته نیز دارای «کارت‌های خط» (Line Card) هستند که معمولاً چهار تا هشت خط ورودی از یک نوع شبکهٔ معین در آنها تعبیه شده است. یک **«کارت خط اترنت»** (Ethernet Line Card) نمی‌تواند مثلاً فریمهای شبکهٔ توکن رینگ را بپذیرد چرا که نمی‌داند در کجای سرآیند فریم آدرس مقصد را پیدا کند! با این وجود یک «پل» می‌تواند برای انواع شبکه‌های مختلف با سرعت متفاوت، «کارتهای خط» مجزا داشته باشد. در یک «پل» برخلاف هاب، هر خط «حوزهٔ تصادم» خاص خود را دارد. «سوئیچها» شبیه به پلها هستند چرا که هر دوی آنها براساس آدرسهای درون فریم، آنها را مسیریابی و هدایت می‌کنند. در واقع بسیاری از افراد این دو واژه را به صورت معادل به کار می‌برند. تفاوت اصلی آنها در این است که یک سوئیچ شبیه به شکل ۴-۴۷-ج، برای وصل کردن کامپیوترهای منفرد به یکدیگر، کاربرد دارد. طبعاً وقتی ماشین میزبان A در شکل ۴-۴۷-ب می‌خواهد فریمی را برای ماشین B بفرستد، پل اگر چه فریم را دریافت می‌کند ولی آن را نادیده می‌گیرد. برخلاف آن در شکل ۴-۴۷-ج، سوئیچ باید بلافاصله فریم را از A به سمت B هدایت نماید چرا که هیچ راهی برای تحویل فریم به B [جز از طریق سوئیچ] نیست. از آنجایی که معمولاً هر یک از پورتهای یک سوئیچ به یک کامپیوتر منفرد وصل می‌شود لذا یک سوئیچ باید پورتهای بسیار بیشتری (در مقایسه با پلهائی که برای وصل تعداد کمی LAN طراحی شده‌اند) داشته باشد. در ضمن هر یک از «کارت‌های خط» بایستی فضای بافر کافی برای ذخیرهٔ فریمهای دریافتی از هر یک از پورتها در اختیار داشته باشند. از آنجائی هر یک از پورتها «حوزهٔ تصادم» مجزا و متعلق به خود را دارند لذا هیچ فریمی در اثر تصادم از بین نخواهد رفت. با این وجود اگر فریمها با نرخی بیش از ظرفیت سوئیچ، وارد گردند ممکن است فضای بافر پر شده و سوئیچ مجبور به حذف آنها شود.

برای آنکه این مشکل کمی کاهش یابد در سوئیچهای مدرن به محض آنکه فیلد آدرس مقصد از فریم وارد گردید، عمل هدایت و انتقال فریم شروع می‌شود، حتی قبل از آنکه مابقی فریم بطور کامل دریافت شده باشد. البته در صورتی که خط خروجی مورد نظر آزاد باشد. این سوئیچها روش «ذخیره و هدایت» (Store & Forward) را به کارنمی‌برند. [در روش «ذخیره و هدایت» ابتدا کل فریم دریافتی ذخیره شده و سپس عملیات هدایت آغاز می‌شود]. این نوع از سوئیچها که اغلب به نام Cut-Through Switches (سوئیچهای میانبر) مشهورند، کاملاً به صورت سخت‌افزاری پیاده سازی می‌شوند در حالیکه پلها عموماً دارای یک CPU واقعی بوده و عمل «ذخیره و هدایت» را توسط نرم‌افزار انجام می‌دهند. بهر حال چون تمام پلها و سوئیچهای مدرن دارای تراشهٔ مدار مجتمع ویژه‌ای جهت هدایت و انتقال فریمها هستند لذا امروزه تفاوت‌های بین پل و سوئیچ بیشتر به مسائل بازاری بستگی دارد تا مسائل فنی.

تا اینجا تکرارکننده‌ها و هابها را بررسی کرده‌ایم که کاملاً شبیه به هم هستند و همنطور به پلها و سوئیچها پرداختیم که آنها نیز شباهت زیادی به یکدیگر دارند. حال به سوی «مسیریاب» (Router) حرکت خواهیم کرد که با تمام ابزارهای فوق تفاوت دارد. وقتی بسته‌ای به یک مسیریاب وارد می‌شود ابتدا سرآیند و فیلدهای انتهائی فریم حذف شده و سپس بستهٔ جاسازی شده در درون فیلد حمل دادهٔ فریم (Payload) (که در شکل ۴-۴۶ به صورت خاکستری نشان داده شده)، تحویل نرم‌افزار مسیریابی می‌شود. این نرم‌افزار برای انتخاب خط خروجی از مشخصات واقع در سرآیند بسته استفاده می‌کند. در بسته‌های IP، سرآیند هر بسته شامل یک آدرس ۳۲ بیتی

(در IPv4) یا ۱۲۸ بیتی (در IPv6) است ولی آدرس ۴۸ بیتی 802 ندارد. نرم‌افزار مسیریابی نمی‌تواند آدرس فریمها را ببیند و حتی نمی‌تواند متوجه شود که بسته از طریق یک LAN دریافت شده یا از روی یک خط نقطه به نقطه. در فصل پنجم مسیریابها و فرآیند مسیریابی را تشریح خواهیم کرد.

در لایهٔ بالاتر به «دروازه‌های انتقال» (Transport Gateway) بر می‌خوریم. این دروازه‌ها ارتباط دو کامپیوتر را که از پروتکلهای اتصال‌گرای متفاوتی در لایهٔ انتقال استفاده می‌کنند، برقرار می‌نماید. به عنوان مثال فرض کنید کامپیوتری که از پروتکل اتصال‌گرای TCP/IP استفاده کرده، می‌خواهد با کامپیوتری که از پروتکل اتصال‌گرای ATM استفاده می‌کند، محاوره نماید. یک دروازهٔ انتقال می‌تواند بسته‌هائی که از طریق یک «اتصال» دریافت می‌شوند را پس از تغییرات لازم بر روی «اتصال» دیگر بفرستد.

در نهایت، «دروازه‌های کاربرد» (Application Gateway) قالب و محتوای داده‌ها را تشخیص می‌دهند و یک «پیام» را به پیامی دیگر ترجمه می‌کنند. بعنوان مثال یک «دروازه پست الکترونیکی» می‌تواند یک پیام اینترنتی [نامهٔ الکترونیکی] را به پیام SMS برای گوشی‌های تلفن همراه ترجمه نماید.

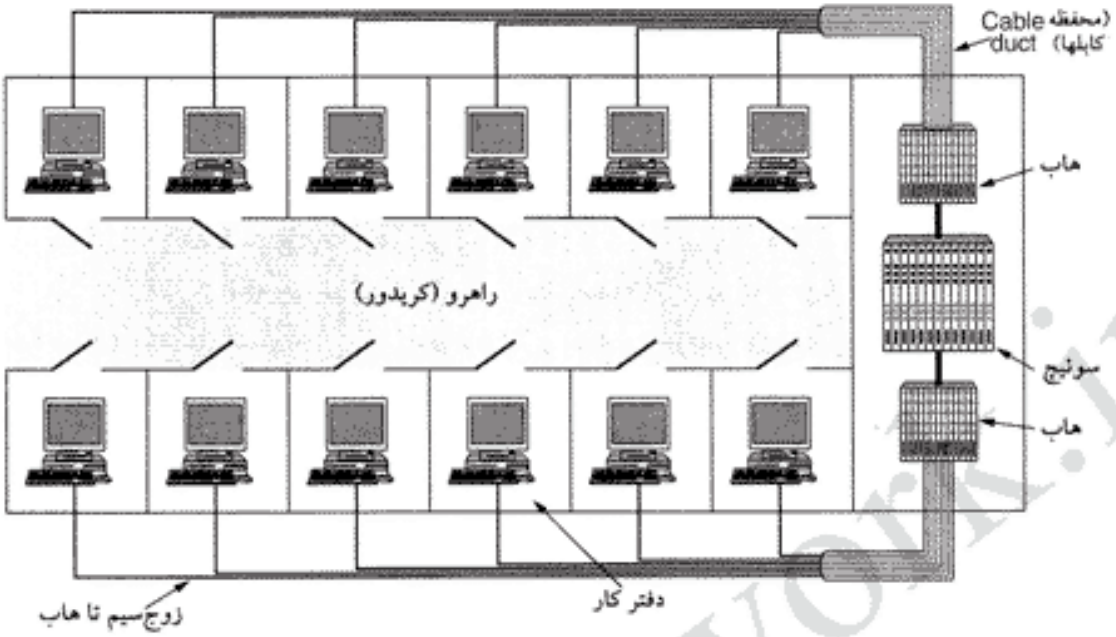
٤ـ۷ـ۶ شبکه‌های محلی مجازی (Virtual LANs)

در دوران اولیه به کارگیری شبکه‌های محلی، کابلهای زردرنگ ضخیم از طریق مجراهای مخصوص (duct)، بین دفاتر ساختمانها کشیده می‌شد و هر کامپیوتری که مسئولین امر تصمیم می‌گرفتند با وصل به این کابل، به شبکه ملحق می‌شد. اغلب، کابلهای بی شماری وجود داشت که به یک «**ستون فقرات مرکزی**» (Central Backbone) یا به یک هاب مرکزی متصل می‌شد (شکل ۴-۳۹) و این موضوع که کدام کامپیوتر به کدام LAN متصل می‌شود چندان مهم نبود. تمام افرادی که در دفاتر مجاور هم بودند بر روی یک LAN مشابه قرار می‌گرفتند، فارغ از آنکه کارشان ارتباطی با هم داشت یا نه! یعنی «**منطق برتر جغرافیائی**». [بعبارت دیگر به جای آنکه یک منطق علمی، ساختار و اعضای یک LAN را تبیین کند جغرافیای محیط این ساختار و اعضاء را مشخص می‌کرد.]

با ابداع 10Base-T و هابها در اویل ۱۹۹۰ همه چیز عوض شد. همهٔ ساختمانها از نو سیم‌کشی شدند (البته با صرف هزینهٔ قابل توجه) تا تمام کابلهای زرد رنگ با قطر شیلنگ باغبانی برچیده شود و به جای آنها از هر دفتر یک زوج‌سیم به یک جعبه تقسیم مرکزی واقع در انتهای راهروها یا یک اتاق مرکزی کشیده شود. (شکل ۴-۴۸ را ببینید.) اگر معاون رئیس یا مسئول سیم‌کشی بلند پرواز بود سیم‌های زوجی ردهٔ ۵ (Cat 5) نصب می‌کرد ولی اگر تنگ نظر بود از سیم‌کشی موجود خطوط تلفن، استفاده می‌نمود (که باز هم چند سال بعد و با پدیدار شدن اترنت سریع باید عوض می‌شد!)

در اترنت مبتنی بر هاب (و بعداً مبتنی بر سوئیچ)، اغلب این امکان وجود داشت که LANها به جای پیکربندی جغرافیایی به صورت **منطقی** پیکربندی شوند. اگر شرکتی به k شبکهٔ LAN نیاز داشته باشد، k عدد هاب خریداری می‌کند. با دقت به آنکه کدام کابل رابط باید به کدام هاب متصل شود اعضای هر LAN را می‌توان به گونه‌ای انتخاب کرد که با ساختار سازمانی آنها مطابقت داشته باشد بدون آنکه جغرافیای محلّ، تاثیر چندانی در این انتخاب بگذارد. البته اگر دو نفر از یک دپارتمان مشابه از سازمان، در ساختمانهای متفاوتی مستقر بودند احتمالاً به دو هاب متفاوت و طبعاً به دو LAN متفاوت متصل می‌شدند. با این حال، شرایط جدید خیلی بهتر از زمانی است که اعضای یک LAN صرفاً بر اساس جغرافیای محلّ تعیین شوند.

آیا این مسئله که چه کسی برروی کدام شبکهٔ LAN قرار گرفته اهمیت دارد؟ مگر نه اینست که سرانجام در هر سازمان تمام LANها به هم متصل می‌شوند؟ کوتاه سخن آنکه جواب مثبت است و مسئله فوق اغلب مواقع اهمیت دارد. مسئولان شبکه بدلائل مختلف علاقمندند که کاربران را به نحوی برروی LAN گروه‌بندی کنند که گروه‌ها به جای آنکه منعکس کنندهٔ نقشه فیزیکی ساختمانها باشند، جلوه‌ای از ساختار سازمانی باشند. یکی از



شکل ۴-۴۸. یک ساختمان با سیم‌کشی مرکزی با بهره‌گیری از هاب و سونیچ.

موارد و دلائل، «امنیت» است. هر کارت شبکه می‌تواند در حالت «بی‌قید» (Promiscuous) قرار بگیرد و تمام ترافیک جاری برروی کانال را دریافت نماید. بسیاری از دپارتمانها همانند دپارتمان پژوهش، ثبت و حسابداری، اطلاعاتی را در اختیار دارند که نمی‌خواهند به بیرون از دپارتمان خودشان راه پیدا کند. در چنین شرایطی قرار دادن تمام افراد برروی یک LAN واحد و جلوگیری از خروج ترافیک از آن LAN، معقول به نظر می‌رسد. مدیریت سازمان تمایلی به شنیدن آنکه «چنین آرایشی ممکن نیست» ندارند مگر آنکه تمام افراد هر دپارتمان، در دفاتر همجوار جای داده شده باشند و در کار یکدیگر فضولی نکنند!!

مورد دیگر «**میزان بار**» است: برخی از LANها نسبت به بقیه، زیادتر مورد استفاده قرار می‌گیرند و ممکن است که تفکیک آنها مطلوبتر باشد. به عنوان مثال اگر گروه پژوهش در حال اجرای انواع آزمایشاتی باشند که گاه ترافیکی بیش از اندازه تولید و شبکهٔ LAN را اشباع می‌کند، گروه حسابداری ممکن است علاقمند نباشد که برای کمک به آنان بخشی از ظرفیت [پهنای باند] خود را وقف آنان کند!

مورد سوّم، «پخش فراگیر» (Broadcasting) است. اغلب LANها از ارسال فراگیر حمایت می‌کنند و بسیاری از پروتکلهای لایه‌های بالاتر از این ویژگی در سطح گسترده‌ای استفاده می‌کنند. به عنوان مثال وقتی کاربری می‌خواهد بسته‌ای برای یک ماشین با آدرس IP معادل X بفرستد چگونه آدرس MAC آن ماشین را بدست می‌آورد تا در فریم مربوطه قرار بدهد؟ ما پاسخ این پرسش را در فصل پنجم بررسی خواهیم کرد ولی اگر بخواهیم بطور خلاصه جمع‌بندی نمائیم پاسخ آنست که ماشین فریمی را به صورت پخش فراگیر برروی شبکه قرار می‌دهد که حاوی این سوال است: «چه کسی صاحب آدرس IP معادل با X است؟»... سپس منتظر پاسخ باقی می‌ماند. نمونه‌های کاربردی زیادی می‌توان یافت که متکی به پخش فراگیر هستند. هر چه LANهای بیشتری به یکدیگر متصل شوند تعداد فریمهای فراگیر که به هر ماشین وارد می‌شوند به صورت خطی و متناسب با تعداد ماشینها افزایش خواهد یافت.

یکی دیگر از مشکلات پخش فراگیر آنست که اگر زمانی یک کارت شبکه از عملکرد طبیعی خود خارج شده و شروع به تولید جریان بی‌پایانی از فریمهای فراگیر نماید تکلیف چیست. نتیجهٔ به پا شدن این «**طوفان فریمهای**

فراگیر» آنست که (۱) کل ظرفیت LAN با این فریمها، اشغال و تباه می‌شود. (۲) تمام ماشینهای واقع در LANهای متصل به هم، به واسطهٔ صرف زمان جهت پردازش و سپس حذف این فریمهای فراگیر، زمین‌گیر می‌شوند.

در بدو امر ممکن است به نظر برسد که «طوفان فریمهای فراگیر» را می‌توان با جدا کردن LANها توسط پل یا سوئیچ محدود کرد ولی اگر هدف نهایی «شفافیت» باشد سوئیچها و پلها موظف به هدایت فریمهای پخش فراگیر هستند. (به عبارتی یک ماشین باید بتواند به یک LAN متفاوت تغییر موقعیت بدهد و هیچکس متوجه این موضوع نشود و در محل جدید نیز قادر به پخش فراگیر باشد.)

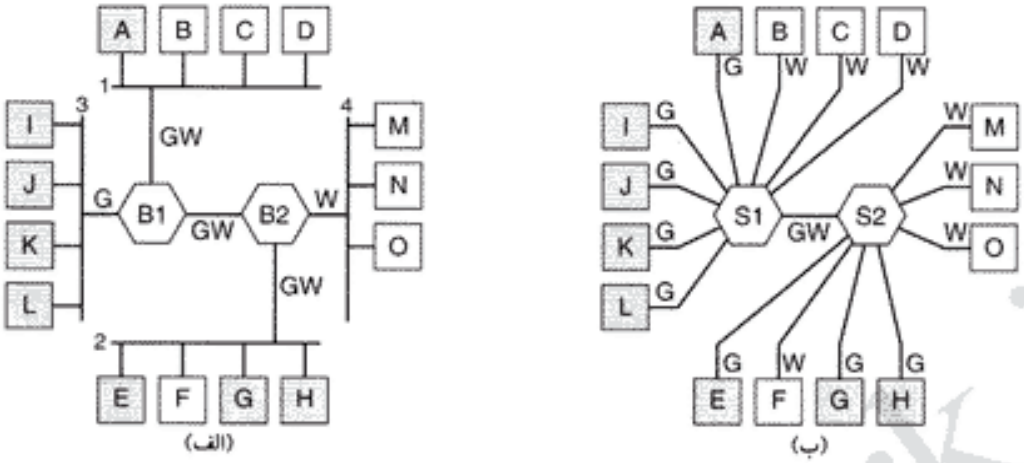
پس از آگاهی از آنکه چرا شرکتها ممکن است بخواهند LANهای متعددی با وسعت محدود داشته باشند، اجازه بدهید به مسئله اصلی یعنی جداکردن توپولوژی منطقی از توپولوژی فیزیکی بپردازیم. فرض کنید که یک کاربر در یک شرکت بدون تغییر محل دفتر کار خود از یک واحد اداری به واحد دیگر منتقل می‌گردد یا برعکس، دفتر کار خود را بدون جابجایی از واحد قبلی خود تغییر می‌دهد. در سیم‌کشی مبتنی بر هاب، تغییر موقعیت به یک LAN جدید مستلزم آنست که مسئول شبکه به سوی جعبه‌تقسیم (Wiring Closet) رفته و کابل رابط ماشین آن کاربر را از محل فعلی بیرون کشیده و آنرا به هاب جدید متصل کند.

در بسیاری از شرکتها تغییر و تحولات سازمانی یک امر عادی است و این مستلزم آنست که مسئول شبکه وقت بسیار زیادی را صرف جدا کردن کابلهای رابط و قرار دادن آن در محلّ جدید نماید. البته در برخی از حالات نیز چنین تغییراتی به هیچ وجه ممکن نیست چرا که مثلاً فاصلهٔ ماشین کاربر از هاب جدید بسیار دور است.

در پاسخ به نیاز کاربران به قابلیت انعطاف بیشتر، عرضه‌کنندگان محصولات شبکه کار برروی طرحی را آغاز کرده‌اند که بر اساس آن بتوان سیم‌کشی ساختمانها را به صورت «نرم‌افزاری» تغییر داد[یعنی پیکربندی ماشینهای هر LAN فارغ از ساختار اتصال فیزیکی آنها ممکن باشد.] نتیجهٔ چنین نظریه‌ای **«شبکهٔ محلّی مجازی»** یا VLAN نامیده می‌شود و توسط کمیته IEEE استانداردسازی شده است. اکنون بسیاری از سازمانها آرایشی مبتنی بر VLAN دارند. اجازه بدهید نگاهی به آن بیندازیم. برای کسب آگاهی بیشتر مراجع مفیدند. (Breyer and Riley , 1999, Seifert , 2000)

شبکه‌های VLAN، بکمک سوئیچهای خاصّ و سازگار با VLAN ، پیاده سازی می‌شوند، هر چند ممکن است همانند شکل ۴-۴۸ دارای چند هاب جانبی و معمولی نیز باشند. برای پیکربندی شبکهٔ مبتنی بر VLAN، مسئول شبکه تصمیم می‌گیرد که (۱) چند VLAN باید تعریف شود. (۲) چه کامپیوترهایی بر روی هر VLAN قرار می‌گیرند. (۳) هر یک از VLANها چگونه نامگذاری می‌شوند. اغلب، یک VLAN با رنگها نامگذاری می‌شود، (البته به صورت غیررسمی) زیرا بدین ترتیب امکان آنکه بتوان نقشه فیزیکی ماشینهای شبکه را به صورت رنگی چاپ کرد، وجود دارد. اعضای شبکهٔ LAN قرمز با رنگ قرمز مشخص می‌شوند، اعضای شبکه LAN سبز با رنگ سبز و به همین ترتیب. با این روش نقشهٔ منطقی و فیزیکی شبکه در یک نمای واحد دیده می‌شود.

به عنوان یک مثال، چهار LAN نشان داده شده در شکل ۴-۴۹ را مد نظر قرار بدهید که در آن هشت ماشین، متعلق به شبکه VLAN خاکستری (G) هستند، هفت تا متعلق به سفید (W) . چهار LAN فیزیکی نیز توسط دو پل B1 و B2 بهم متصل شده‌اند. اگر در این شکل به جای ساختار باس از هابهای مرکزی مبتنی بر زوج سیم استفاده می‌شد ممکن بود که چهار هاب نیز وجود داشته باشد (که در شکل نشان داده نشده است) ، ولیکن از دیدگاه منطقی کابلهای چنداتصالی و هاب مشابه به هم هستند. اگر می‌خواستیم شکل را با هاب ترسیم کنیم ، گویائی خود را از دست می‌داد. امروزه واژه پل بیشتر اشاره به ساختاری شبیه به شکل ۴-۴۹-ب دارد که در آن چندین ماشین از طریق پورتهای آن به هم متصل شده‌اند در حالیکه امروزه «پل» و «سوئیچ» مفهوم معادلی دارند. در شکل



شکل ۴-۴۹. (الف) چهار LAN فیزیکی با استفاده از دو پل، دو VLAN خاکستری و سفید تشکیل داده‌اند.(ب) همان پانزده ماشین بکمک دو سوئیچ، دو VLAN تشکیل داده‌اند.

۴-۴۹-ب همان تعداد ماشین و همان تعداد VLAN [معادل با شکل ۴-۴۹-الف] فقط با استفاده از سوئیچ نشان داده شده است بگونه‌ای که بر روی هر پورت سوئیچ فقط یک ماشین وجود دارد.

برای آنکه VLAN بدرستی کار کند بایستی یک جدول پیکربندی درون هر پل یا سوئیچ تنظیم شود. این جدول مشخص‌کنندهٔ آنست که از طریق کدام یک از پورتها می‌توان به کدام VLAN دسترسی داشت. وقتی فریمی از روی یک VLAN مثلاً خاکستری به سوئیچ وارد شود صرفاً باید بر روی تمام پورتهائی که علامت G دارند منتقل شوند. همانند ترافیک معمولی تک‌پخشی (Unicast) ، ارسال ترافیک چندپخشی (Muticast) و ارسال ترافیک فراگیر (Broadcast) نیز به همین نحو ممکن خواهد بود.

دقت کنید که ممکن است یک پورت برچسب رنگی چندین VLAN را داشته باشد. در شکل ۴-۴۹-الف چنین ساختاری را مشاهده می‌نمائیم. فرض کنید ماشین A یک فریم «فراگیر» [(Broadcast)را برای همهٔ اعضای VLAN خود] ارسال کند. پل B1 این فریم را دریافت کرده و متوجه می‌شود که توسط ماشینی بر روی VLAN خاکستری تولید شده است لذا آن فریم را برروی تمام پورتهائی که علامت G دارند (به استثنای پورتی که فریم از روی آن دریافت شده) ارسال می‌نماید. از آنجایی که B1 فقط دو پورت دیگر با برچسب G دارد لذا فریم را برروی هر دوی آنها ارسال می‌کند.

در B2 داستان به گونهٔ دیگری است: در اینجا پل می‌داند که هیچ ماشین خاکستری برروی LAN 4 وجود ندارد لذا فریم برروی آن هدایت نخواهد شد و فقط به LAN 2 ارسال می‌شود. اگر یکی از کاربران LAN 4 ملزم به تغییر واحد اداری خود شده و به VLAN خاکستری نقل مکان نماید باید جدول B2 بهنگام‌سازی شده و تنها پورت آن که بابرچسب W مشخص شده به GW تغییر داده شود. اگر ماشین F به شبکهٔ مجازی خاکستری نقل مکان کند، پورت متصل به LAN 2 باید از GW به G تغییر برچسب داده شود.

حال فرض کنید ماشینهای عضو LAN 2 و LAN 4 همگی به جمع خاکستری‌ها بپیوندند، لذا نه تنها پورتهای پل B2 که متصل به LAN 2 و LAN 4 هستند علامت G می‌گیرند بلکه پورت اتصال B1 به B2 نیز از GW به G تغییر علامت می‌دهد چراکه لازم نیست فریمهای سفیدی که از LAN 3 یا LAN 1 می‌رسند به B2 هدایت شوند. در شکل ۴-۴۹-ب همین وضعیت حاکم است و تمام پورتهایی که به یک ماشین واحد متصل هستند با یک برچسب تک‌رنگ مشخص می‌شوند چراکه هر ماشین تنها به یک VLAN متعلق است.

تاکنون فرض را بر آن گذاشته بودیم که پلها و سوئیچها به نحوی می‌دانند که رنگ یک فریم ورودی چیست.

آنها چگونه از این موضوع آگاه می‌شوند؟ برای این کار سه روش زیر کاربرد دارد:

۱. به هر پورت رنگ VLAN انتساب داده شود.
۲. به هر آدرس MAC یک رنگ VLAN منتسب گردد.
۳. به آدرسهای لایهٔ ۳ یا آدرس IP ماشین یک رنگ VLAN منتسب شود.

در روش اوّل، به هر پورت یک برچسب رنگ داده می‌شود که این برچسب، VLAN مربوطه را مشخص می‌کند. با این حال این روش فقط زمانی کار خواهد کرد که تمام ماشینهای متصل به آن پورت، به یک VLAN مشابه متعلق باشند. در شکل ۴-۴۹-الف این ویژگی برای پورت بین پل B1 و LAN 3 صادق است در حالیکه برای پورت متصل به LAN 1 صادق نیست.

در روش دوّم، پل یا سوئیچ دارای جدولی است که در آن فهرست آدرسهای ۴۸ بیتی تمام ماشینهای متصل به آن (یعنی آدرس MAC) و مشخصات VLAN هر ماشین ، درج می‌شود. تحت این شرایط، می‌توان برروی یک LAN فیزیکی (مثل LAN 1 در شکل ۴-۴۹-الف) چند VLAN مختلط تعریف کرد. وقتی فریمی دریافت می‌شود، آنچه که هر پل یا سوئیچ باید انجام بدهد آنست که آدرس MAC آن را استخراج و درون جدول به دنبال آن بگردد و ببیند که فریم از کدام VLAN می‌آید.

در روش سوّم، پل یا سوئیچ موظف است تا محتوای فیلد حمل داده (Payload) از هر فریم را بررسی کرده و به عنوان مثال تمام ماشینهای مبتنی بر IP را در یک VLAN و ماشینهای مبتنی بر Apple Talk را در VLAN دیگر دسته‌بندی کند. در حالت اوّل می‌توان برای تشخیص هویت هر ماشین از آدرس IP آن استفاده کرد. این استراتژی زمانی بسیار سودمند خواهد بود که تعداد بی‌شماری از ماشینهای شبکه کامپیوترهای کیفی هستند و می‌توانند در هر یک از مکانهای متعدد و مجاز قرار بگیرند. از آنجایی که هر ایستگاه آدرس MAC خاص خود را دارد لذا با توقف ایستگاه در محل جدید، آدرس MAC آن نمی‌تواند چیزی در مورد آن VLAN که ایستگاه عضو آنست ، مشخص کند.

تنها مشکل این روش آنست که یکی از قوانین اساسی در شبکه بندی را نقض می‌کند: «استقلال لایه‌ها» تشخیص آنکه چه چیزی در فیلد حمل داده از فریم قرار گرفته برعهدهٔ لایهٔ پیوند داده‌ها نیست. این لایه نباید محتوای داده‌های درون هر فریم را بررسی کند و یا براساس محتوای درون آن تصمیمی بگیرد. یکی از تبعات به کارگیری این روش آنست که هرگاه در پروتکل لایهٔ ۳ تغییری ایجاد شود (مثلاً IPv4 به IPv6 ارتقاء پیدا کند) سوئیچ بلافاصله از کار خواهد افتاد. متاسفانه سوئیچهایی که بدین نحو کار می‌کنند به وفور در بازار عرضه شده‌اند.

البته تعریف VLAN بر اساس آدرس IP، اشکالی در مسیریابی بسته‌های IP بوجود نخواهد آورد ولیکن تلفیق وظائف لایه‌ها سرآغاز بروز مشکلات پیش‌بینی نشده خواهد بود. (تقریباً کلّ فصل پنجم به مسیریابی IP اختصاص دارد). شاید عرضه‌کنندگان سوئیچ چنین استدلالی را به مسخره بگیرند و بگویند سوئیچهای آنه IPv4 و IPv6 را به رسمیت می‌شناسند و همه چیز به درستی پیش خواهد رفت ولیکن اگر زمانی IPv7 مطرح شد چه اتفاقی می‌افتد؟ شاید فروشنده بگوید که در آن زمان سوئیچهای جدید بخرید!! آیا این کار خیلی شاقّ است؟

استاندارد IEEE 802.1Q

اندکی اندیشهٔ بیشتر در خصوص VLAN ما را به این نتیجه می‌رساند که آنچه واقعاً اهمیت دارد آنست که هر فریم ارسالی نام VLAN خود را با خود حمل کند نه آنکه VLAN ماشین فرستنده را سوئیچ بروشهای دیگری مشخص کند. اگر روشی برای مشخص کردن VLAN در سرآیند هر فریم وجود داشته باشد دیگر نیازی به بررسی داده‌های

درونی هر فریم نخواهد بود. برای شبکه‌های جدیدی مثل 802.11 یا 802.16، اضافه کردن فیلد VLAN به سرآیند هر فریم نسبتاً ساده است. در حقیقت فیلد Connection Identifier در 802.16 ذاتاً چیزی شبیه به VLAN Identifier است. ولی در مورد اترنت چه کاری می‌توان کرد؟ شبکه‌ای که رایجترین شبکهٔ دنیاست و هیچ فیلد اضافی برای تعبیهٔ VLAN Identifier در آن، تعریف نشده است.

کمیتهٔ IEEE 802 این مسئله را در دستور کار خود قرار داد و پس از مباحثات فراوان کاری غیرقابل تصور انجام داد و سرآیند اترنت را عوض کرد. قالب جدید فریم، در استاندارد IEEE 802.1Q تدوین و در سال ۱۹۹۸ منتشر شد. در قالب جدید، هر فریم دارای یک برچسب VLAN است (با نام VLAN Tag) که مختصراً آن را بررسی خواهیم کرد. متاسفانه تغییر در استانداردی مثل سرآیند اترنت که کاملاً جا افتاده و بطور رایج از آن استفاده می‌شود، چندان ساده نیست. در این خصوص ممکن است سوالات زیر به ذهن خطور کند:

۱. آیا باید چندین میلیون کارت اترنت موجود را دور انداخت؟

۲. اگر نه، چه کسی فیلدهای جدید را تولید نماید؟

۳. برای فریمهایی که اندازهٔ حداکثر دارند چه اتفاقی می‌افتد؟ [چون نمی‌توان به فریمی که بر طول حداکثر آن محدودیت گذاشته شده، داده‌ای افزود.]

البته کمیتهٔ 802 از این مشکلات آگاه بود و می‌بایست راه حلّی مناسب ارائه می‌کرد، کاری که نهایتاً انجام شد.

نکتهٔ اصلی در راه حل ارائه شده آنست که فیلدهای VLAN فقط توسط پلها و سوئیچها مورد استفاده قرار می‌گیرد و ماشینهای کاربران بدان نیازی ندارند. بدین ترتیب در شکل ۴-۴۹ وقتی فریمها مستقیماً به سوی یک ایستگاه پایانی ارسال می‌شوند به این فیلدها نیازی نیست و فقط روی خطوطی بین پلها یا سوئیچها، بکار می‌آیند. بنابراین برای بکارگیری VLAN، این پلها یا سوئیچها هستند که باید VLAN را به رسمیت بشناسند؛ این موضوع نیز در پلها و سوئیچها از قبل پیش‌بینی شده و یک نیاز تلقی می‌شود. حال باید نیازمندیهای جدیدی را معرفی نمائیم که 802.1Q بدانها پاسخ داده است.

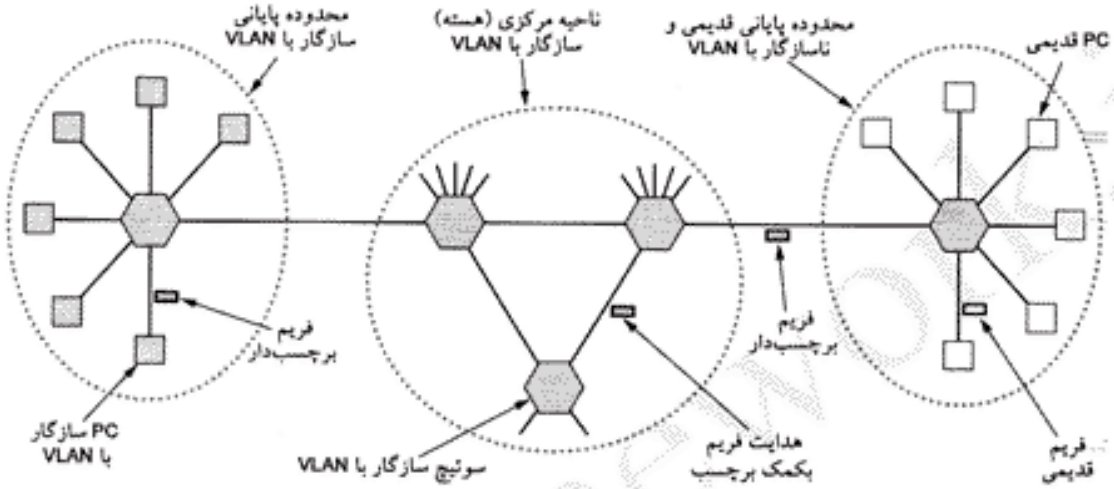
پاسخ به این سوال که «آیا باید تمام کارتهای اترنت موجود را دور انداخت؟» منفی است. به خاطر داشته باشید که کمیتهٔ 802.3 حتی نتوانست به افراد بقبولاند که فیلد Type را به فیلد Length تغییر بدهند. حال شما تصور کنید عکس‌العمل مردم در قبال اعلام دور انداختن کارتهای اترنت، چه می‌شد! البته وقتی کارتها جدید به بازار بیاید شاید بتوان امیدوار بود که سازگار با 802.1Q بوده و فیلدهای مربوط به VLAN در آنها تعبیه شده باشند.

به هر حال اگر کارت شبکهٔ تولیدکنندهٔ یک فریم، فیلدهای VLAN را بدان اضافه نکند، چه کسی باید این کار را انجام بدهد؟ پاسخ این سوال آنست که اولین پل یا سوئیچ مبتنی بر VLAN که فریم را دریافت می‌کند، این فیلدها را به فریم افزوده و آخرین آنها در مسیر، این فیلدها را حذف می‌نماید. ولیکن یک سوئیچ یا پل از کجا می‌فهمد که یک فریم به کدام VLAN تعلق دارد؟ پاسخ آنست که اولین پل یا سوئیچ می‌تواند به هر یک از پورتهای خود یک شمارهٔ VLAN نسبت بدهد یا آنکه به آدرس MAC آن فریم نگاه کند و یا محتوای داده‌های درونی را بررسی کند (کار ممنوع!).

در خصوص کارتهای اترنت که سازگار با 802.1Q هستند، مشکلی وجود ندارد. انتظار قابل تحقق آنست که تمام کارتهای اترنت گیگابیت از همان ابتدا سازگار با 802.1Q باشند و با ارتقاء کارتهای قدیمی به اترنت گیگابیت، 802.1Q نیز به صورت خودکار جای خود را باز کند. برای حلّ این مشکل که فریمهای اترنت نباید از 1518 بایت بیشتر باشند در 802.1Q طول حداکثر به ۱۵۲۲ بایت افزایش یافته است.

در دوران گذار از اترنت فعلی به اترنت گیگابیت، در بسیاری از شبکه‌ها برخی از ماشینهای قدیمی (عموماً

اترنت کلاسیک و اترنت سریع) با VLAN سازگار نیستند، در حالیکه برخی دیگر (عموماً اترنت گیگابیت) از آن پشتیبانی می‌کنند. چنین وضعیتی در شکل ۴-۵۰ به تصویر کشیده شده است و در آن نمادهای خاکستری سازگار با VLAN و نمادهای بی‌رنگ ناسازگار هستند. برای سادگی بحث، فرض را بر آن گذاشته‌ایم که تمام سوئیچها با VLAN سازگار هستند. اگر اینگونه نباشد اوّلین سوئیچ سازگار با VLAN بکمک آدرس MAC یا آدرس IP درون فریم، برچسب لازم را بدان خواهد افزود.



شکل ۴-۵۰. گذر از اترنت قدیمی به اترنت سازگار با VLAN. نمادهای خاکستری با VLAN سازگارند؛ نمادهای بی‌رنگ سازگار نیستند.

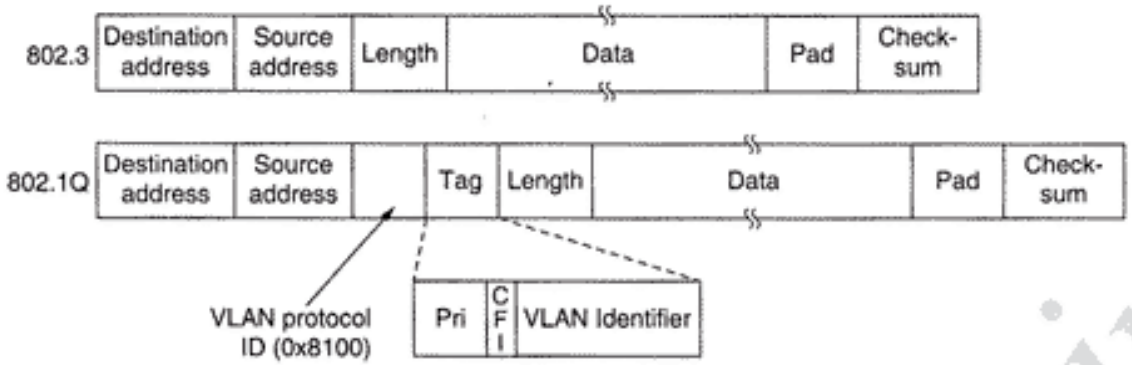
در این شکل کارتهای اترنت سازگار با VLAN ، مستقیماً فریمهای برچسب‌دار VLAN (یعنی فریمهای 802.1Q) را تولید می‌نمایند و سوئیچهای بعدی از این برچسب استفاده خواهند کرد. برای عملیات هدایت فریمها (یعنی عمل سوئیچنگ) هر سوئیچ بایستی بداند که کدام VLAN از طریق کدام پورت در دسترس است. (طبق توضیحات قبلی) دانستن آنکه فریم جاری به VLAN مثلاً خاکستری تعلق دارد فایدهٔ چندانی نخواهد داشت مگر آنکه سوئیچ بداند کدام پورتها به ماشینهای عضو VLAN خاکستری متصل هستند. بنابراین سوئیچ نیازمند به داشتن جدولی اندیس‌دار و مرتب شده بر حسب مشخصهٔ VLAN است که تعیین می‌کند از کدامیک از پورتها باید استفاده شود و آیا ماشینهای متصل بدین پورتها سازگار با VLAN هستند یا قدیمیند.

وقتی یک PC قدیمی فریمی را برای یک سوئیچ سازگار با VLAN می‌فرستند، سوئیچ مربوطه با استفاده از دانش قبلی[1] خود در خصوص VLAN متناظر با فرستندهٔ فریم (مثلاً با استناد به شمارهٔ پورت فیزیکی، آدرس MAC یا آدرس IP) برچسب لازم را به فریم چسبانده و فریم جدیدی تولید می‌نماید. از این دیدگاه، دیگر قدیمی بودن ماشین فرستنده هیچ اهمیتی ندارد. به روش مشابه هرگاه نیاز باشد فریمی برچسب‌دار (مبتنی بر 802.1Q) تحویل ماشینی قدیمی شود قبل از تحویل، ساختار فریم به قالب بدون برچسب تبدیل خواهد شد.

حال بیائید نگاهی به قالب فریم 802.1Q بیندازیم. قالب این نوع فریم در شکل ۴-۵۱ نشان داده شده است. تنها تغییر، اضافه شدن یک جفت فیلد ۲ بایتی (جمعاً ۴ بایت) به قالب قدیمی است. یکی از آنها فیلد مشخصه VLAN Protocol ID است که همیشه مقدار 0x8100 دارد. از آنجایی که این مقدار بیش از ۱۵۰۰ است تمام

---

۱. به خاطر داشته باشید که سوئیچهای مدرن سازگار با VLAN ، بصورت نرم‌افزاری پیکربندی می‌شوند و بعضاً دارای سیستم عامل و فرامین مخصوص هستند. بنابراین در خصوص ماشینهای باکارت اترنت قدیمی تنظیمات VLAN ، باید بصورت دستی انجام شود.-م



شکل ۴ـ۵۱. قالب فریم قدیمی اترنت 802.3 و فریم 802.1Q.

کارتهای اترنت آنرا به عنوان فیلد «نوع فریم» (Type) تفسیر می‌کنند و به فیلد طول داده Length تعبیر نخواهد شد. عکس‌العملی که کارتهای اترنت قدیمی در برخورد با چنین فریمهایی از خود نشان می‌دهند جای بحث دارد چراکه فرض بر آن است که چنین فریمهایی برای کارتهای قدیمی ارسال نمی‌شود.

فیلد دو بایتی بعدی شامل سه زیرفیلد است که اصلی‌ترین آنها یعنی VLAN Identifier (شناسهٔ VLAN)، ۱۲ بیت کم‌ارزش را به خود اختصاص داده است. این فیلد کل آن چیزی است بدان نیازمند بوده‌ایم: یعنی مشخص می‌کند که «این فریم به کدام VLAN تعلق دارد». فیلد ۳ بیتی Priority (اولویت) فعلاً کاری در خصوص VLAN بر عهدهٔ ندارد و عملاً بلااستفاده است. استدلال کمیتهٔ 802.1Q در تعریف این زیرفیلد آن بوده که چون تغییر در سرآیند اترنت فرآیند دشوار و زمانبری است ، حالا که این کار خطیر میسر شده چرا چیزهای خوب دیگر نیز بدان اضافه نشود؟ این فیلد ۳ بیتی این امکان را فراهم آورده تا بتوان «ترافیک بی‌درنگ از نوع سخت و نرم» (Soft & Hard Realtime Traffic) [مثل صدا و تصویر یا نظائر آن] را از ترافیک غیرحساس به زمان [مثل ترافیک ارسال نامه‌های الکترونیکی] تشخیص داد و امکان ارائه «کیفیت خدمات» (QoS) در اترنت نیز میسر شود. در آینده به این فیلد جهت ارسال صدا بر روی اترنت نیاز خواهد شد. (هر چند مشابه با همین فیلد در پروتکل IP ربع قرن قبل در نظر گرفته شده بود ولیکن هیچگاه از آن استفاده نشد!!)

آخرین زیرفیلد، بیت CFI (Canonical Format Indicator) است که در حقیقت می‌بایست بیت CEI (Corporate Ego Indicator) بمعنای مشخصهٔ شخصی شرکت نامیده می‌شد. اساساً این بیت باید برای مشخص کردن آدرسهای Big Endian از آدرسهای Little Endian بکار می‌رفت ولی پس از بحث و جدل فراوان از آن صرفنظر شد.[1] اکنون معنای این بیت آن است که درون فیلد حمل داده از فریم اترنت، یک فریم کامل از نوع 802.5 (فریم شبکهٔ Token Ring) قرار دارد و انتظار می‌رود به یک شبکهٔ محلی از نوع 802.5 دیگر تحویل شود چراکه اتصال آنها از طریق اترنت برقرار شده است. (به عبارت دیگر این فیلد مشخص می‌کند که شبکهٔ اترنت باید به عنوان حامل میانی، یک فریم نوع 802.5 را به شبکه‌ای دیگر از همین نوع تحویل بدهد.)[2] البته این بیت هیچ کاری در مورد شبکهٔ VLAN انجام نمی‌دهد ولیکن سیاستهای کمیته استانداردسازی IEEE متفاوت با سیاستهای عادّی است!!!

با دقت در توضیحات فوق، وقتی یک فریم برچسب‌دار به یک سوئیچ سازگار با VLAN می‌رسد آن سوئیچ از

---

۱. معنای آدرس Big Endian آن است که بایتهای پرارزش آدرس در ابتدا و بایتهای کم‌ارزش در انتها قرار گرفته و به همان نحو ارسال می‌شوند. معنای آدرس Little Endian آن است که بایتهای کم‌ارزش در ابتدا و بایتهای پر ارزش آدرس بعد از آن ارسال می‌شوند.

۲. یعنی نوعی از پدیدهٔ تونل‌زنی (Tunneling) در سطح لایهٔ پیوند داده. -م

فیلد VLAN ID به عنوان اندیس (یا بعبارتی کلید جستجو) در جدول خود استفاده کرده و به کمک آن پورتهایی که این فریم باید بر روی آنها فرستاده شود را مشخص می‌کند. ولی سوال اساسی اینجاست که این جدول از کجا بدست می‌آید؟ اگر این جدول باید به صورت دستی تنظیم شود به پلهٔ شمارهٔ صفر برگشته‌ایم یعنی «پیکربندی دستی پلها» ؛ زیبائی پلهای شفاف در آن است که به محض وصل ، فوراً به کار می‌افتند (یعنی Plug & Play است) و نیازی به پیکربندی دستی ندارند. از دست دادن این ویژگی بسیار ناگوار است! خوشبختانه پلهای سازگار با VLAN می‌توانند خود را بر اساس برچسب فریمهای ورودی به صورت خودکار پیکربندی کنند. مثلا اگر فریمی با برچسب VLAN 4 از روی پورت شماره ۳ وارد شود ، به روشنی مشخص است که ماشین متصل به پورت ۳ متعلق به VLAN 4 است. استاندارد 802.1Q روش ایجاد پویای این جداول را تشریح کرده ولی در اکثر جاهای این توضیحات خواننده را به بخشهایی خاصّ از الگوریتم پرلمن ارجاع داده است؛ الگوریتم پرلمن نیز در 802.1D استاندارد شده است.

قبل از خاتمهٔ بحث مسیریابی در VLAN ، اشاره به یک نکتهٔ پایانی خالی از لطف نیست. بسیاری از افراد در دنیای اینترنت و اترنت طرفدار ساختار شبکه‌های «بی‌اتصال» (Connectionless) هستند و در مقابل هر چیزی که زمینه نیاز به «اتصال» در لایهٔ «پیوند داده» یا لایهٔ شبکه را فراهم کند به شدت مقاومت می‌کنند. در VLAN چیزهایی معرفی شده که شبیه به مفهوم «اتصال» هستند. برای عملکرد صحیح VLAN هر فریم با خود یک مشخصه خاص و جدید دارد [یعنی همان VLAN ID] که از آن بعنوان اندیس جستجو در جدول درون سوئیچ ، استفاده می‌شود. این دقیقاً همان اتفاقی است که در شبکه‌های اتصال‌گرا می‌افتد. در شبکه‌های بدون‌اتصال به تنها چیزی که برای مسیریابی و هدایت فریم نیاز است آدرس MAC می‌باشد و به هیچ نوع مشخصهٔ اتصال یا امثال آن نیاز نیست. در فصل پنجم در خصوص مفاهیم اتصال‌گرائی (Connectionism) بیشتر خواهیم آموخت.

۸.٤ خلاصه

برخی از شبکه‌ها تنها دارای یک کانال انتقال مشترک هستند که همهٔ ایستگاهها از آن، برای مبادله داده‌های خود استفاده می‌کنند. در این شبکه‌ها، مسئله اصلی در طراحی لایهٔ پیوند داده ، تسهیم و تخصیص کانال بین ایستگاه‌های رقیبی است که علاقمندند از آن استفاده نمایند. الگوریتمهای بی‌شماری برای مسئله تخصیص کانال ابداع شده است. خلاصه‌ای از مهمترین روشهای تخصیص کانال در جدول ۴-۵۲ فهرست شده است.

ساده‌ترین روش تخصیص، FDM و TDM است. این روشها زمانی مفیدند که تعداد ایستگاه‌ها کم و ثابت و ترافیک آنها پیوسته باشد. هر دوی این روشها در محیطهایی با این ویژگی مثلاً برای تقسیم پهنای باند در خطوط اصلی تلفن (شاهراه‌ها)، کاربرد دارند. وقتی تعداد ایستگاه‌ها زیاد و متغیر یا ترافیک آنها نسبتاً انفجاری است روش‌های TDM و FDM گزینه‌های مناسبی نیستند. پروتکل ALOHA (چه نوع معمولی و چه نوع Slotted آن) بعنوان روشهای جایگزین TDM و FDM معرفی شدند. گونه‌های مختلف ALOHA نیز تشریح و تحلیل شده‌اند. برخی از این گونه‌ها امروزه در سیستمهای واقعی به کار می‌روند.

وقتی بتوان حالت فعلی کانال را شنود (احساس) کرد، ایستگاه‌ها قادر خواهند بود مادامیکه ایستگاه دیگر در حال ارسال است از ارسال فریم خود اجتناب کنند. این تکنیک یعنی **«شنود سیگنال حامل»** منتج به ابداع پروتکلهای متنوعی شد که می‌تواند در شبکه‌های LAN و MAN به کار گرفته شود.

رده‌ای از پروتکلها قادرند مسئله رقابت بر سر تصرف کانال را منتفی کنند یا حداقل این رقابت را تا حَد قابل توجهی کاهش بدهند. روش «شمارش دودوئی معکوس» (Binary Countdown) رقابت را بطور کامل حذف می‌کند. پروتکل «پیمایش درخت وفقی» (Adaptive Tree Walk) با تقسیم‌بندی پویای ایستگاه‌ها به گروه‌های از

روش	توصیف عملکرد
FDM	به هر ایستگاه یک باند فرکانسی تخصیص می‌دهد.
WDM	الگوی پویای FDM برای فیبر نوری
TDM	به هر ایستگاه یک برش زمان تخصیص می‌دهد.
Pure ALOHA	ارسال ناهماهنگ در هر لحظه دلخواه
Slotted ALOHA	ارسال تصادفی در برشهای زمانی مشخص
1-persistent CSMA	استاندارد دسترسی چندگانه مبتنی بر شنود کانال
Nonpersistent CSMA	تاخیر تصادفی در هنگامی که کانال اشغال تشخیص داده می‌شود.
P-persistent CSMA	همان CSMA است با این تفاوت که احتمال اصرار در ارسال فریم P است.
CSMA/CD	همان CSMA است با این تفاوت که بمحض تشخیص تصادم ادامه ارسال را قطع می‌کند.
Bit map	زمان‌بندی به نوبت و چرخشی (Round-Robin) بکمک یک الگوی بیتی (Bitmap)
Binary countdown	ایستگاه با بزرگترین شماره حق ارسال در نوبت بعدی را دارد.
Tree walk	رقابت محدود با فعالیتهای انتخابی (تعریف گروه‌های رقیب در ساختار درختی)
MACA, MACAW	پروتکلهای شبکه‌های محلی بی‌سیم
Ethernet	همان CSMA/CD با الگوریتم عقبگرد نمایی
FHSS	جهشهای فرکانسی در طیف گسترده (Frequency Hopping Spread Spectrum)
DSSS	Direct Sequence Spread Spectrum
CSMA/CA	استاندارد دسترسی چندگانه مبتنی بر شنود کانال و اجتناب از تصادم

شکل ۴-۵۲. روشها و سیستمهای تخصیص یک کانال مشترک.

هم جدا، مشکل رقابت را بطور جدی کاهش می‌دهد. تقسیم‌بندی گروه‌ها حتی‌الامکان به نحوی است که فقط گروههایی که در آنها فقط یک ایستگاه فریمی برای ارسال آماده دارد مجاز به ارسال هستند.

شبکه‌های محلی بی‌سیم مشکلات و راه حل‌های خاص خود را دارند. عمده‌ترین مشکل توسط ایستگاه‌های «پنهان» ایجاد می‌شود و بدین دلیل CSMA جوابگوی چنین شبکه‌هایی نخواهد بود. رده‌ای از این راه‌حلها که در طبقهٔ MACA و MACAW دسته بندی شده‌اند سعی دارند ایستگاه‌ها را وادار به ارسال در پیرامون ماشین مقصد نمایند تا بدین نحو CSMA عملکرد بهتری داشته باشد. روش‌های مبتنی بر طیف گسترده از نوع Frequency Hopping و Direct Sequence نیز در این شبکه‌ها به کار گرفته شده است. IEEE 802.11 روش CSMA و MACAW را تلفیق کرد و روش CSMA/CA را معرفی نمود.

اترنت رایجترین نوع شبکه‌های محلی است که برای تخصیص کانال از روش CSMA/CD بهره می‌گیرد. نسخه قدیمی آن از یک کابل واحد که بین تمام ماشینها کشیده می‌شد، استفاده می‌کردند، در حالیکه اکنون استفاده از هاب، سوئیچ و کابلهای زوجی بسیار رایج است. سرعت آن از ۱۰ مگابیت‌برثانیه تا یک گیگابیت بر ثانیه افزایش یافته و هنوز هم در حال افزایش است.

امروز شبکه‌های محلی بی‌سیم نیز در حال رواج هستند که در این بین استفاده از 802.11 وسیعتر است. لایهٔ فیزیکی در این شبکه، امکان پنج نوع انتقال و مدولاسیون متفاوت را فراهم آورده است که شامل مادون قرمز، گونه‌های متفاوت مبتنی بر طیف گسترده و «سیستم FDM چندکاناله» است. این شبکه می‌تواند یک ایستگاه ثابت در هر سلول داشته باشد ولیکن قادر است بدون ایستگاه ثابت نیز کار کند. این پروتکل گونه‌ای از MACAW و متکی بر شنود مجازی سیگنال حامل است.

شبکه‌های بین شهری بی‌سیم (Wireless MAN) در حال پیدایش و رواج هستند. این گونه شبکه‌ها

سیستمهای باند گسترده‌ای هستند که از امواج رادیوئی بهره می‌گیرند تا جایگزین ارتباطات تلفنی شوند. در آنها از تکنیک‌های مدولاسیون باند باریک استفاده شده است. کیفیت خدمات نیز از موارد مهمی است که استاندارد 802.16 چهار ردهٔ مختلف از این گونه خدمات را تعریف کرده است: ارسال با نرخ ثابت ، دو روش ارسال با نرخ متغیر و روش ارسال مبتنی بر بهترین تلاش (Best Effort) .

سیستم بلوتوث نیز بی‌سیم است ولیکن هدف آن وصل ابزارهای رومیزی است؛ مثلا برای اتصال گوشیها و ابزارهای جانبی کامپیوترهای شخصی بدون نیاز به سیم کاربرد دارد. همچنین به منظور وصل ابزارهایی مثل دورنگار (فکس) و تلفن‌های همراه به کار می‌آید. بلوتوث شبیه به 802.11 از تکنیک طیف گسترده مبتنی بر پرش فرکانس بهره گرفته و درباند ISM [یعنی 2.4 Ghz] کار می‌کند. به خاطر بالا بودن سطح نویز در بسیاری از محیطها و نیاز به فعل و انفعال بی‌درنگ، در پروتکلهای مختلف آن از روش تصحیح خطای مستقیم (Forward Error Correction) بهره گرفته شده است.

بدلیل وجود انواع متفاوت LAN به روشی جهت اتصال آنها به یکدیگر نیاز است. از پل و سوئیچ به همین منظور استفاده می‌شود. در پلهای نوع Plug & Play از «الگوریتم درخت پوشا» (Spanning Tree) استفاده شده است. پیشرفت جدید در دنیای شبکه‌ها، VLAN است که بکمک آن توپولوژی منطقی LANها از توپولوژی فیزیکی آن جدا می‌شود. قالب جدیدی برای فریمهای اترنت معرفی شده است تا راحتر بتوان VLAN را در سازمان‌ها پیاده سازی کرد.

مسائل

۱. برای حل این مسئله از رابطه‌ای که در همین فصل آمده بهره بگیرید ولیکن در ابتدا آن را بیان نمائید. بطور تصادفی فریمهایی برای انتقال بر روی کانالی با نرخ ارسال 100 Mbps تولید می‌شوند. اگر در زمان تولید یک فریم کانال اشغال بود تا رسیدن نوبت به آن در صف منتظر خواهد ماند. طول فریمها دارای تابع توزیع نمائی با میانگین 10000 bits/Frame است. برای هر یک از نرخهای تولید فریم که در زیر مشخص شده، تاخیری را که هر فریم (با طول متوسط) با آن مواجه می‌شود (شامل زمان انتظار صف و زمان انتقال) حساب نمائید.

الف) ۹۰ فریم در ثانیه

ب) ۹۰۰ فریم در ثانیه

ج) ۹۰۰۰ فریم در ثانیه

۲. یک گروه Nتائی از ایستگاه‌ها دارای کانالی مشترک با نرخ 56kbps هستند و روش تخصیص کانال نیز Pure ALOHA است. بطور متوسط هر ایستگاه در هر صد ثانیه یک فریم هزار بیتی تولید می‌کند (حتی اگر فریم قبلی هنوز ارسال نشده باشد چراکه ایستگاه‌ها می‌توانند فریمهای خروجی را در بافر خود نگاه دارند). مقدار حداکثر N چقدر می‌تواند باشد؟

۳. تاخیر Pure ALOHA را در مقایسه با روش Slotted ALOHA و در شرایط بار پائین مدّ نظر قرار بدهید. تاخیر کدامیک کمتر است؟ پاسخ خود را تشریح کنید.

٤. ده هزار ایستگاه رزرو بلیط هواپیما، برای استفاده از یک کانال واحد، بروش Slotted ALOHA با هم رقابت می‌کنند. هر ایستگاه بطور متوسط ۱۸ تقاضا در هر ساعت خواهد داشت. برشهای زمانی (Time Slot) ۱۲۵ میکروثانیه‌ای هستند. مقدار تقریبی بار کّل کانال چقدر است؟

۵. جمع کثیری از کاربران سیستم ALOHA در هر ثانیه ۵۰ تقاضا تولید می‌نمایند (که این مقدار شامل تعداد

تقاضاهای جدید و تقاضاهای ارسال مجدد فریمهایی است که قبلا در اثر تصادم خراب شده‌اند). زمان به برشهای متساوی ۴۰میلی‌ثانیه‌ای تقسیم شده است:

الف) احتمال موفقیت ارسال در همان دفعه اول چقدر است؟

ب) احتمال بروز دقیقاً k تصادم پیاپی و سپس یک موفقیت چقدر است؟

ج) به طور متوسط برای ارسال یک فریم چند بار تلاش لازم است؟

۶. اندازه‌گیری بار کانال Slotted ALOHA با تعداد نامحدود کاربر، نشان می‌دهد که ده درصد از برشهای زمانی بلااستفاده مانده‌اند:

الف) بار کانال یعنی G چقدر است؟

ب) بازده مفید کانال (Throughput) چقدر است؟

ج) آیا بار کانال بیش از اندازه بالاست یا کمتر از حد متعادل است؟

۷. در یک سیستم Slotted ALOHA با کاربران نامحدود، متوسط تعداد برش زمانی که هر ایستگاه بین تصادم و ارسال مجدّد صبر می‌کند، ۴ است. منحنی «تاخیر» را بر حسب «بازده مفید کانال» (Delay Versus throughput) در این سیستم ترسیم نمائید.

۸. یک ایستگاه مثل S را در شبکه‌ای با پروتکلهای زیر در نظر بگیرید. این ایستگاه قبل از ارسال فریمش بر روی کانال ، (در بدترین حالت) چه مدت زمانی باید انتظار بکشد:

الف) پروتکل Basic Bit Map

ب) پروتکلهای Mok و Wark با جایگشت مجازی شمارهٔ ایستگاه‌ها

۹. یک LAN از پروتکل «شمارش دودوئی معکوس» نسخهٔ Mok و Wark ، بهره می‌گیرد. در یک لحظهٔ خاص، ده ایستگاه دارای شماره‌های مجازی ۸، ۲، ۴، ۵، ۱، ۷، ۳، ۶، ۹ و ۰ هستند. سه ایستگاه ارسال‌کنندهٔ بعدی به ترتیب عبارتند از: ۴، ۳ و ۹. پس از آنکه این ایستگاه‌ها ارسال خود را به اتمام رساندند شماره مجازی ایستگاه‌ها چند خواهد بود؟ [ترتیب شماره‌ها را از راست به چپ در نظر بگیرید.]

۱۰. شانزده ایستگاه که از ۱ تا ۱۶ شماره‌گذاری شده‌اند برای استفاده از یک کانال اشتراکی بروش «پیمایش درخت وفقی» رقابت می‌کنند. اگر تمام ایستگاه‌هایی که آدرس آنها اعداد اول است بطور ناگهانی آماده ارسال شوند برای خاتمه مراحل رقابت چند «برش بیتی» (Bit Slots) نیاز خواهد بود؟

۱۱. مجموعه‌ای از 2^n ایستگاه برای دسترسی به یک کابل اشتراکی از روش «پیمایش درخت وقفی» بهره گرفته‌اند. در یک زمان خاصّ دو ایستگاه آماده ارسال می‌شوند. اگر $1<<2^n$ باشد مقدار حداقل، حداکثر و میانگین تعداد برشهای زمانی برای پیمایش این درخت [و خاتمهٔ رقابت] چقدر است؟

۱۲. در شبکهٔ محلی بی‌سیم که مطالعه کردیم به جای استفاده از پروتکل CSMA/CD از پروتکلی نظیر MACA استفاده شده بود. تحت چه شرایطی (در صورت امکان) استفاده از روش CSMA/CD میسر خواهد بود؟

۱۳. چه ویژگیهایی از پروتکلهای دسترسی به کانال در GSM و WDMA مشترک هستند؟ GSM را در فصل ۲ مرور نمائید.

۱۴. شش ایستگاه A تا F با بکارگیری پروتکل MACA با یکدیگر در ارتباط و تبادل داده هستند. آیا امکان انتقال همزمان اطلاعات در این شبکه وجود دارد؟ (پاسخ خود را شرح بدهید.)

۱۵. در هر طبقه از یک ساختمان هفت طبقه، ۱۵ دفتر کار همجوار وجود دارد. در هر دفتر یک پریز دیواری (Wall Socket) برای وصل یک پایانه در جلوی آن تعبیه شده است و بدین ترتیب پریزها یک مستطیل در چهارگوش یک صفحه قائم تشکیل داده‌اند که فاصله بین پریزها به صورت عمودی و افقی چهار متر است.