بسم الله
الرحمن
الرحيم

دانشگاه آزاد اسلامی

واحد تهران جنوب

دانشکده مدیریت و حسابداری

پایان‌نامه برای دریافت درجه کارشناسی ارشد "M.A."

(رشته مدیریت صنعتی– گرایش تحقیق در عملیات)

عنوان:

بهینه‌سازی دسترس‌پذیری سیستم k ازn تعمیرپذیر با استفاده از الگوریتم رقابت استعماری

زمستان ۱۳۹۳

دانشگاه آزاد اسلامی

واحد تهران جنوب

دانشکده مدیریت و حسابداری

پایان نامه برای دریافت درجه کارشناسی ارشد *"M.A."*

(رشته مدیریت صنعتی – گرایش تحقیق در عملیات)

عنوان :

بهینه سازی دسترس پذیری سیستم‌های k از n تعمیر پذیر با استفاده از الگوریتم رقابت استعماری

تقدیم به

پدر و مادرم بزرگ ترین حامیان من در زندگی که

بودن و ماندنم از آن ها بوده و هست

فهرست مطالب

عنوان مطلب | شماره صفحه

چکیده ۱

۱ فصـــل اول: کلیـــات تحـقیـق..... ۲

۱-۱. مقدمه ۳

۱-۲. شرح مسأله ٤

۱-۳. ضرورت و اهمیت مسأله ٦

۱-٤. اهداف تحقیق ٦

۱-۵. سوالات تحقیق ۷

۱-۵-۱ . سوالات اصلی ۷

۱-۵-۲ . سوالات فرعی ۷

۱-٦. فرضیه تحقیق ۷

۱-۷. نوع روش تحقیق ۷

۱-۸. روش گردآوری اطلاعات ۷

۱-۹. ابزار گردآوری اطلاعات ۷

۱-۱۰. تعاریف واژه‌ها و اصطلاحات فنی و تخصصی ۸

۱-۱۰-۱ . قابلیت اطمینان سیستم ۸

۱-۱۰-۲ . سیستم تعمیرپذیر ۸

۱-۱۰-۳ . سیستم‌های غیرقابل تعمیر (تعمیرناپذیر) ۸

۱-۱۰-٤ . دسترس‌پذیری ۸

۱-۱۰-۵ . سیستم‌های K از N ۸

۱-۱۰-٦ . فرآیند مارکوف ۹

۱-۱۰-۷ . زنجیره‌های مارکوف ۹

۱-۱۰-۸ . مسائل NP-HARD ۹

۱-۱۰-۹ . الگوریتم رقابت استعماری ۹

۲ فصـــل دوم : مروری بر مبـانی و پیشینه تحـقیـق..... ۱۱

۲-۱. مقدمه ۱۲

۲-۲. تعاریف، مفاهیم و شاخص‌های قابلیت اطمینان ۱۲

۲-۲-۱ . دسترس پذیری ۱۳
۲-۲-۲ . قابلیت نگهداری ۱٤
۲-۲-۳ . ایمنی ۱٤
۲-۲-٤ . ماندگاری ۱۵
۲-۲-۵ . پایداری ۱۵
۲-۲-٦ . قابلیت دوام ۱٦
۲-۳. مدل سازی شبکه و ارزیابی قابلیت اطمینان انواع سیستم‌ها ۱٦
۲-۳-۱ . مفاهیم مدل سازی شبکه ۱٦
۲-۳-۱-۱ . سیستم، زیرسیستم، واحد ۱٦
۲-۳-۱-۲ . سیستم‌های تعمیرپذیر ۱۷
۲-۳-۱-۳ . سیستم‌های تعمیر ناپذیر ۱۷
۲-۳-۲ . ریاضیات آنالیز قابلیت اطمینان ۱۷
۲-۳-۲-۱ . نرخ خرابی یا نرخ شکست ۱۸
۲-۳-۲-۲ . میانگین زمان خرابی ۱۹
۲-۳-۲-۳ . میانگین زمان بین دو خرابی ۲۰
۲-۳-۳ . انواع سیستم‌ها در ارزیابی قابلیت اطمینان ۲۱
۲-۳-۳-۱ . سیستم‌های با شبکه سری (متوالی) ۲۱
۲-۳-۳-۲ . سیستم‌های با شبکه موازی ۲٤
۲-۳-۳-۳ . سیستمهای با شبکه **K** از **N** ۲۷
۲-۳-۳-۳-۱ . سیستم **K** از **N** با توزیع طول عمر مستقل و همسان و تعمیر ناپذیر ۲۸
۲-۳-۳-۳-۲ . سیستم **K** از **N** با توزیع طول عمر مستقل، ناهمسان و تعمیر ناپذیر ۲۹
۲-۳-۳-۳-۳ . ارزیابی تصادفی سیستم **K** از **N** با اجزای دارای طول عمر مستقل و همسان ۳۰
۲-۳-۳-۳-٤ . ارزیابی تصادفی سیستم **K** از **N** با اجزای دارای طول عمر مستقل و ناهمسان ۳۱
۲-۳-۳-۳-۵ . سیستم‌های **K-OUT-OF-N** تعمیرپذیر ۳۱
۲-٤. تئوری صف – زنجیره های مارکف ۳۲
۲-٤-۱ . فرآیند مارکف ۳۲
۲-٤-۲ . زنجیره های مارکف ۳۳
۲-٤-۳ . ماتریس گذار ۳۳
۲-٤-٤ . گذار **M** مرحله‌ای ۳٤

۲-٤-٥ . احتمالات حدی در زنجیره مارکف ۳۵
۲-٤-٦ . زنجیره های مارکف با زمان پیوسته ۳۷
۲-٤-۷ . ماتریس گذار و ماتریس آهنگ گذار ۳۸
۲-٤-۸ . نمودار آهنگ ۳۹
۲-٤-۹ . روابط حدی در زنجیره مارکف با زمان پیوسته ٤۰
۲-٤-۱۰ . فرآیند تولد و مرگ ٤۲
۲-٤-۱۱ . بیان فرآیند تولد و مرگ در چارچوب زنجیره مارکوف ٤۳
۲-٤-۱۲ . معادلات تعادلی در سیستم های نمایی ٤٤
۲-٥. الگوریتم رقابت استعماری ٤٥
۲-٥-۱ . مقدمه ٤٥
۲-٥-۲ . تعریف رقابت استماری ٤٦
۲-٥-۳ . مروری تاریخی بر پدیده استعمار ٤۷
۲-٥-٤ . الگوریتم رقابت استعماری ٤۹
۲-٥-٤-۱ . .ایجاد جواب های اولیه (شکل دهی امپراتوری‌های اولیه) ٥۱
۲-٥-٤-۲ . مدل‌سازی سیاست جذب(حرکت مستعمره‌ها به سمت امپریالیست) ٥٤
۲-٥-٤-۳ . جا به جایی موقعیت مستعمره و امپریالیست ٥٦
۲-٥-٤-٤ . قدرت کل یک امپراتوری ٥۷
۲-٥-٤-٥ . .رقابت استعماری ٥۷
۲-٥-٤-٦ . سقوط امپراتوری‌های ضعیف ٦۰
۲-٥-٤-۷ . همگرایی ٦۱
۲-٥-٥ . کاربردها ٦۱
۲-٦. پیشینه تحقیق ٦۳
۲-٦-۱ . پیشینه داخلی ٦۳
۲-٦-۲ . پیشینه خارجی ٦٥
۲-۷. جمع بندی ۷۰
۳ فصـــــل ســوم : روش شنـــاسی تحـقیـق ۷۱
۳-۱. مقدمه ۷۲
۳-۲. نوع تحقیق ۷۲
۳-۳. روش تحقیق ۷۲

۳-٤. روش جمع آوری اطلاعات ۷۳
۳-٥. روش تجزیه و تحلیل اطلاعات ۷۳
۳-٦. مدل ریاضی سیستم K از N تعمیرپذیر ۷٤
۳-٦-۱ . فرضیات مدل ۷٤
۳-٦-۲ . نمادها ۷٤
۳-٦-۳ . معادلات تعادلی و دسترس پذیری سیستم K از N تعمیر پذیر ۷٥
۳-٦-٤ . مدل مسأله ۷۷
۳-٦-٥ . مدل های عددی مسأله ۸۲
۳-٦-٥-۱ . مثال ۱ ..... ۸۲
۳-٦-٥-۲ . مثال ۲ ..... ۸٤
۴ فصـــل چهارم: یـافتـــه های تحـقیـق ۸۶
٤-۱. مقدمه ۸۷
٤-۲. حل دقیق مسأله ۸۷
٤-۲-۱ . نتایج حل دقیق مثال ۱ ..... ۸۷
٤-۲-۲ . نتایج حل دقیق مثال ۲ ..... ۸۸
٤-۳. حل توسط الگوریتم فرا ابتکاری ۸۹
٤-۳-۱ . نتایج حل مثال ۱ توسط الگوریتم رقابت استعماری ۸۹
٤-۳-۲ . نتایج حل مثال ۲ توسط الگوریتم رقابت استعماری ۸۹
٤-٤. تحلیل و مقایسه ۹۰
٤-٤-۱ . تحلیل زمان صرف شده ۹۰
٤-٤-۲ . تحلیل کارایی و همگرایی الگوریتم رقابت استعماری پیشنهادی ۹۱
۵ فصـــل پنجم: نتیـجه گیری و پیـشنهادات ۹۴
٥-۱. مقدمه ۹٥
٥-۲. نتیجه گیری ۹٥
٥-۳. پاسخ به سوالات تحقیق ۹٦
٥-٤. میزان دستیابی به اهداف تحقیق ۹۷
٥-٥. پیشنهادات آتی ۹۷
۶ منـــابع ۹۹
٦-۱. مقاله ۱۰۰

٦-٢. کتاب ١٠١
٦-٣. پایان نامه و گزارش های علمی ١٠٢
٦-٤. مجموعه مقالات همایشها ١٠٢
٦-٥. منابع الکترونیکی ١٠٢
٧ پیـــــوســـــت ١٠٣
٧-١. کد حل دقیق مثال ١ توسط نرم افزار متلب ١٠٤
٧-٢. کد حل دقیق مثال ٢ توسط نرم افزار متلب ١٠٦
٧-٣. کد الگوریتم رقابت استعماری برای مثال ١ ..... ١٠٨
٧-٣-١. ICA.M ١٠٨
٧-٣-٢. COST1.M ١١٠
٧-٣-٣. SHARESETTINGS.M ١١٢
٧-٣-٤. CREATEINITIALEMPIRES.M ١١٢
٧-٣-٥. ASSIMILATECOLONIES.M ١١٤
٧-٣-٦. DOREVOLUTION.M ١١٤
٧-٣-٧. INTRAEMPIRECOMPETITION.M ١١٥
٧-٣-٨. UPDATETOTALCOST.M ١١٥
٧-٣-٩. INTEREMPIRECOMPETITION.M ١١٦
٧-٣-١٠. ROULETTEWHEELSELECTION.M ١١٧
٧-٤. کد الگوریتم رقابت استعماری برای مثال ٢ ..... ١١٧
٧-٤-١. ICA.M ١١٧
٧-٤-٢. COST1.M ١١٩
٧-٤-٣. SHARESETTINGS.M ١٢١
٧-٤-٤. CREATEINITIALEMPIRES.M ١٢٢
٧-٤-٥. ASSIMILATECOLONIES.M ١٢٣
٧-٤-٦. DOREVOLUTION.M ١٢٤
٧-٤-٧. INTRAEMPIRECOMPETITION.M ١٢٥
٧-٤-٨. UPDATETOTALCOST.M ١٢٥
٧-٤-٩. INTEREMPIRECOMPETITION.M ١٢٥
٧-٤-١٠. ROULETTWHEELSELECTION.M ١٢٦

۷-۵. نتایج حاصل از ۱۰۰ بار اجرای مثال ۱ توسط الگوریتم رقابت استعماری ۱۲۸
۷-٦. نتایج حاصل از ۱۰۰ بار اجرای مثال ۲ توسط الگوریتم رقابت استعماری ۱۳۳

چکیده انگلیسی ۱۳۲

فهرست جداول

عنوان جدول **شماره صفحه**

جدول ۲-۱(رابطه قابلیت اطمینان سیستم سری با تعداد واحدهای آن) ۲۳

جدول ۲-۲ (رابطه قابلیت اطمینان سیستم موازی با تعداد واحدهای آن) ۲۶

جدول ۲-۳ (کاربردهای الگوریتم رقابت استعماری) ۶۱

جدول ۲-٤(پیشینه تحقیقات داخلی و خارجی) ۶۸

جدول ۳-۱(پارامترهای ورودی مثال ۱) ۸۳

جدول ۳-۲ (پارامترهای ورودی مثال ۲) ۸٤

جدول ٤-۱ (نتایج حل دقیق مثال ۱ توسط نرم افزار متلب) ۸۷

جدول ٤-۲(نتایج حل دقیق مثال ۲ توسط نرم افزار متلب) ۸۸

جدول ٤-۳ (نتایج حل مثال ۱ توسط الگوریتم رقابت استعماری) ۸۹

جدول ٤-٤(نتایج حل مثال ۲ توسط الگوریتم رقابت استعماری) ۸۹

جدول ٤-۵ (نتایج آماری بدست آمده از ۱۰۰ تکرار الگوریتم پیشنهادی برای دو مثال) ۹۲

فهرست شکل‌ها

عنوان شکل **شماره صفحه**

شکل ۱-۱ (سیستم سری موازی شامل y زیرسیستم و ساختار k از n) ۵

شکل ۲-۱ ..... ۲۰

شکل ۲-۲ (بلوک دیاگرام سیستم سری با n عضو متوالی) ۲۱

شکل ۲-۳ (نمایش TTF یک سیستم سری) ۲۲

شکل ۲-٤ (بلوک دیاگرام سیستم موازی با n عضو موازی) ۲٤

شکل ۲-۵ (نمایش TTF یک سیستم موازی) ۲٤

شکل ۲-٦ (RBD در ساختار K از N برای سیستم تعمیر ناپذیر) ۲۸

شکل ۲-۷ (ارتباط یک واحد افزونه بجای واحد از کار افتاده در یک سیستم 2-out-of-3) ۲۹

شکل ۲-۸ (گذار سیستم به حالت های مختلف نسبت به زمان در زنجیره مارکف پیوسته) ۳۸

شکل ۲-۹(نمودار آهنگ یک سیستم مارکوف) ٤۰

شکل ۲-۱۰(نمودار آهنگ یک سیستم مارکوف) ٤۱

شکل ۲-۱۱(نمودار آهنگ در فرآیند تولد و مرگ) ٤٤

شکل ۲-۱۲(اعمال سیاست جذب از سوی استعمارگران بر مستعمرات) ٤۹

شکل ۲-۱۳(شبه کد الگوریتم عمومی رقابت استعماری) ۵۰

شکل ۲-۱٤(اجزای اجتماعی سیاسی تشکیل دهنده یک کشور) ۵۱

شکل ۲-۱۵(چگونگی شکل‌گیری امپراتوری‌های اولیه) ۵۳

شکل ۲-۱٦(شمای کلی حرکت مستعمرات به سمت امپریالیست) ۵٤

شکل ۲-۱۷ (حرکت واقعی مستعمرات به سمت امپریالیست) ۵٦

شکل ۲-۱۸(تغییر جای استعمارگر و مستعمره) ۵۷

شکل ۲-۱۹(کل امپراتوری، پس از تغییر موقعیت‌ها) ۵۷

شکل ۲-۲۰ (شمای کلی رقابت استعماری) ۵۸

شکل ۲-۲۱ (سقوط امپراتوری ضعیف) ٦۰

شکل ۳-۱(مدل تجزیه و تحلیل اطلاعات) ۷۳

شکل ۳-۲(دیاگرام انتقال یک سیستم با ساختار k از n) ۷۵
شکل ۳-۳ (نمودار آهنگ انتقال مدل مسأله) ۷۸
شکل ۳-٤(ساختار سیستم مسأله) ۸۰
شکل ۳-۵(مدل ریاضی مسأله) ۸۱
شکل ۳-٦ (ساختار سیستم در مثال ۱) ۸۳
شکل ۳-۷(ساختار سیستم در مثال ۲) ۸۵

فهرست نمودارها

عنوان نمودار **شماره صفحه**

نمودار ٤-١(زمان صرف شده توسط الگوریتم حل دقیق) ۹۰
نمودار ٤-٢ (زمان صرف شده توسط الگوریتم رقابت استعماری)..... ۹۱
نمودار ٤-٣ (همگرایی الگوریتم رقابت استعماری بکارگرفته شده در مثال ۱)..... ۹۱
نمودار ٤-٤(همگرایی الگوریتم رقابت استعماری بکارگرفته شده در مثال ۲)..... ۹۲

چکیده

طراحی بهینه سیستم و بهینه سازی قابلیت اطمینان نقش کلیدی در طراحی مهندسی و بطور بسیار موثر در افزایش عملکرد سیستم دارد. و از کار افتادن محصول و سیستم‌ها موجب وقوع اختلال در سطوح مختلفی می‌شود و می‌تواند حتی به عنوان تهدیدی شدید برای جامعه و محیط زیست نیز تعلقی شود. بدین دلیل در هر جامعه مدرن، مهندسان و مدیران فنی مسئول برنامه‌ریزی، طراحی، ساخت و بهره‌برداری از ساده‌ترین محصول تا پیچیده‌ترین سیستم‌ها هستند.

یکی از مهم‌ترین مسائل در افزایش قابلیت اطمینان و دسترس‌پذیری سیستم‌ها، طراحی سیستم‌ها بگونه‌ای است که بیشترین قابلیت اطمینان و دسترس‌پذیری را داشته باشند. برای افزایش میزان دسترس پذیری شیوه‌های مختلفی در ادبیات وجود دارد از جمله قابلیت اطمینان خود اجزاء سیستم، افزایش سطح افزونگی (افزایش تعداد N) و همچنین ترکیبی از این دو است.

در این پایان نامه مسأله تخصیص اجزای اضافی برای یک سیستم شامل y زیر سیستم که ساختار هرکدام از آنها k از n است و بصورت سری با هم در ارتباط هستند مورد بررسی قرار گرفته است. با توجه به تعمیرپذیر بودن اجزاء مدل زیر سیستم‌ها، مدل مورد نظر بگونه‌ای است که تعداد اجزای اضافی و تعمیرکاران را باید به‌نحوی مشخص کرد تا دسترسی پذیری سیستم بهینه شود. برای بدست آوردن دسترسی پذیری در این نوع سیستم که تابع هدف مسأله ما است از زنجیره مارکف استفاده شده است. و در ادامه کار به بهینه‌سازی دسترسی پذیری سیستم و در نتیجه بهبود سیستم پرداخته شده است.

تابع هدف، ماکزیمم سازی دسترسی پذیری کل سیستم با توجه به محدودیت‌های وزن، حجم و هزینه است که هزینه شامل دو قسمت، هزینه تعمیرکاران و هزینه خود اجزاء است. و متغیرهای تصمیم گیری تعداد اجزاء و تعمیرکاران مورد استفاده در هر زیر سیستم است.

از آنجا که مسأله ارائه شده یک مسأله از نوع برنامه ریزی غیرخطی عدد صحیح است از الگوریتم رقابت استعماری برای بهینه سازی استفاده شده است. در پایان برای ارزیابی و اعتبار سنجی رویکرد الگوریتم پیشنهادی، یک مقایسه میان جواب بدست آمده توسط الگوریتم ارائه شده و حل دقیق انجام شده است.

واژگان کلیدی: دسترسی پذیری، تعمیرپذیری، سیستم k از n، زنجیره مارکف، الگوریتم رقابت استعماری

فصـــــل اول

کلیـــــات تحـــقیـــق

۱-۱. مقدمه

افزایش قابلیت اطمینان[1] و سایر مشخصاتش مانند دسترس‌پذیری[2]، طول عمر[3]، توانایی بازسازی دوباره[4]، ایمنی[5]، ماندگاری[6]، و دیگر مشخصات مهم فنی با توجه به محدودیت‌های هزینه‌ای، بودجه‌ای، وزنی، حجمی و ...، بسیار مهم و کاربردی بوده و همچنین از عوامل موفقیت در بازار رقابتی امروز می‌باشد.

بدین دلیل در هر جامعه مدرن، مهندسان و مدیران فنی مسئول برنامه‌ریزی، طراحی، ساخت و بهره‌برداری از ساده‌ترین محصول تا پیچیده‌ترین سیستم‌ها هستند. از کار افتادن محصول و سیستم‌ها موجب وقوع اختلال در سطوح مختلفی می‌شود و می‌تواند حتی به عنوان تهدیدی شدید برای جامعه و محیط زیست نیز تلقی شود. از این رو مصرف کنندگان و بطور کلی مردم جامعه انتظار دارند که محصول و سیستم‌ها پایا، اطمینان بخش و ایمن باشند. بنابراین به عنوان یک پرسش اساسی چنین مطرح است که قابلیت اطمینان سیستم و سایر مشخصاتش مانند دسترس پذیری و ... در طول عمر کاری آینده‌اش چه میزانی است و یا حتی ایمنی آن چقدر است؟ این پرسشی است که بخش‌هایی از آن را می‌توان با ارزیابی و کمیت سنجی قابلیت اطمینان پاسخ گفت.

شیوه‌های ارزیابی قابلیت اطمینان و مشخصه‌های آن از نظر تاریخچه پیدایش، بدواً در ارتباط با صنایع هوا- فضا و کاربردهای نظامی شکل گرفت ولی سریعاً توسط سایر صنایع مانند صنایع هسته‌ای که تحت فشار شدیدی جهت تضمین ایمنی، دسترس‌پذیری و قابلیت اطمینان راکتورهای هسته‌ای در تامین انرژی الکتریکی می‌باشند. و با صنایع فرآیندهای پیوسته مانند صنایع فولاد و صنایع شیمیایی که هر ساعت از توقف آن‌ها بعلت وقوع معایب می‌تواند موجب تحمیل خسارت‌های بزرگ مالی، جانی و آلودگی محیط زیست شود مورد توجه و کاربرد قرار گرفت.

اگر چه شایان توجه است که در تمام این زمینه‌ها شاهد وقوع مسائل و مشکلات شدیدی در سال‌های اخیر بوده‌ایم. از جمله در صنایع هوا- فضا در سال ۱۹۸٦ حادثه‌ای که برای فضاپیمای چلنجر[7] پیش آمد و همچنین حادثه‌ای دیگر مربوط به برخی هواپیمای تجاری و مسافری. در صنایع هسته‌ای، حادثه‌ی نیروگاه چرنوبیل[8] در سال

---

[1] Reliability
[2] Availability
[3] Longevity
[4] Recoverability
[5] Safety
[6] Survivability
[7] Space Shuttle Challenger
[8] Chernobyl

۱۹۸٦ و جزیره سه مایلی [1]در سال ۱۹۷۹ در تامین انرژی الکتریکی خاموشی نیویورک[2] در سال ۱۹۷۷. در صنایع شیمیایی، حادثه‌های فلیکس برو[3] در سال ۱۹۷٤، سی وی سو[4] در سال ۱۹۷٦، بوپال[5] در سال ۱۹۸٤، و بسیاری از حوادث دیگر، که در تمام این رخدادها خسارت‌های چشمگیر و شدیدی بر جامعه و محیط زیست تحمیل شد.

این رخدادها فشار فزاینده‌ای برای لزوم توجه به ارزیابی قابلیت اطمینان و مشخصه‌های آن وارد کرده و اهمیت بالای این مقوله را نشان می‌دهد(رضائیان، ۱۳۹۳).

۱-۲. شرح مسأله

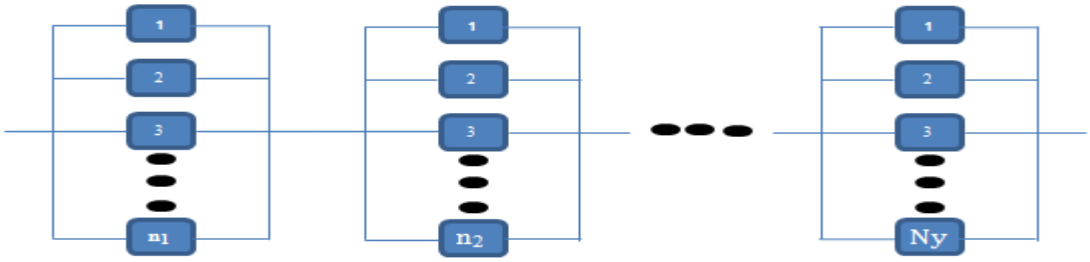
امروزه ساختارهای مختلفی در طراحی سیستم‌ها با توجه به پیشینه بودن قابلیت اطمینان و دسترس پذیری بکار گرفته می‌شود. یکی از پرکاربردترین ساختارها، ساختار اجزای اضافی k از n[6] می‌باشد که در آن برای این که یک سیستم در حال کار باشد باید حداقل k جزء از n جز کار کنند.

یکی از مهم‌ترین مسائل در افزایش قابلیت اطمینان و دسترس‌پذیری سیستم‌ها، طراحی سیستم‌ها بگونه‌ای است که بیشترین قابلیت اطمینان و دسترس‌پذیری را داشته باشند. برای افزایش میزان دسترس پذیری شیوه‌های مختلفی در ادبیات وجود دارد از جمله قابلیت اطمینان خود اجزاء سیستم، افزایش تعداد اجزای اضافی (افزایش تعداد N) و همچنین ترکیبی از این دو است. که یکی از مشهورترین شیوه‌ها برای افزایش قابلیت اطمینان و دسترس‌پذیری سیستم‌ها، افزایش تعداد اجزای اضافی سیستم است. میزان افزایش اجزای اضافی در سیستم با توجه به محدودیت-های سیستم صورت می‌گیرد و به مسأله تخصیص متغیر اضافی[7] نیز شهرت پیدا کرده‌اند.

در این پژوهش مسأله تخصیص متغیر اضافی برای یک سیستم شامل y زیر سیستم که ساختار هر کدام از آن‌ها k از n است و بصورت سری با هم در ارتباط هستند مورد بررسی قرار گرفته است.

---

[1] Three Mile Island
[2] New York City blackout
[3] Flixborough
[4] Seveso
[5] Bhopal
[6] K-out-of-N
[7] Redundancy Allocation



شکل ۱-۱ (سیستم سری موازی شامل y زیرسیستم و ساختار k از n)

با توجه به تعمیرپذیر بودن[1] اجزاء زیر سیستم‌ها، باید تعداد اجزای اضافی و تعداد تعمیرکاران را طوری مشخص کنیم که دسترس‌پذیری سیستم بهینه بشود. برای به‌دست‌آوردن دسترس‌پذیری در این نوع سیستم که تابع هدف مسأله ما می‌باشد از زنجیره مارکف[2] استفاده شده است.

تابع هدف مسأله، ماکزیمم سازی دسترس‌پذیری کل سیستم با توجه به محدودیت‌های وزن، حجم و هزینه می‌باشد که هزینه خود شامل دو قسمت، هزینه تعمیرکاران و هزینه اجزاء بوده است و متغیرهای تصمیم شامل تعداد اجزاء مورد استفاده در هر زیر سیستم و همچنین تعداد تعمیرکاران مورد استفاده در این زیر سیستم‌ها می‌باشد.

از آنجا که مسأله از نوع برنامه‌ریزی غیر خطی عدد صحیح و همچنین تخصیص اجزای اضافی در سیستم سری موازی[3] می‌باشد در دسته‌ی مسایل NP-hard[4] قرار می‌گیرد (Ying shen juang, et al., 2008). لذا تضمینی وجود ندارد که در زمان چند جمله‌ای جواب بهینه برای این مسأله یافت شود. از این رو استفاده از روش‌های فرا ابتکاری[5] یا الگوریتم‌های تکاملی برای یافتن جواب بهینه‌ی این مسأله بهینه‌سازی پیچیده، مناسب تشخیص داده شده است.

در این پژوهش، یکی از جدیدترین و کاراترین روش‌های جستجوی فرا ابتکاری یعنی الگوریتم رقابت استعماری[6] برای بهینه‌سازی استفاده شده است. این الگوریتم روشی در حوزه محاسبات تکاملی است که به یافتن پاسخ بهینه مسائل مختلف بهینه‌سازی می‌پردازد.

---

[1] Repairable

[2] Markov chain

[3] Series-parallel

[4] Non-deterministic Polynomial-time hard

5 Metaheuristic

6 Imperialist Competitive Algorithm (ICA)

۱-۳. ضرورت و اهمیت مسأله

طراحی بهینه سیستم و بهینه سازی قابلیت اطمینان نقش کلیدی در طراحی مهندسی و بطور بسیار موثر در افزایش عملکرد سیستم دارد. سیستم تعمیرپذیر به سیستمی اشاره دارد که می‌تواند بعد از هر خرابی تعمیر شود و بطور مطلوب بعد از هر تعمیر به عملکرد خود ادامه دهد و دسترس‌پذیری مفهومی کاملاً وابسته به قابلیت اطمینان دارد و به مقیاس اندازه‌گیری قابلیت اطمینان سیستم اشاره دارد . برای سیستم تعمیرپذیر، دسترس‌پذیری یک معیار بسیار مهم است. و برای رسیدن به سطح دسترسی مورد نیاز سیستم معمولاً از اجزای اضافی و سیستم‌های تعمیر استفاده می‌شود، و هنگام افزایش تعداد اجزای اضافی و تیم‌های تعمیر به منظور دستیابی به دسترس‌پذیری بالاتر هزینه سیستم نیز افزایش پیدا می‌کند. بنابراین طراحان سیستم و تصمیم‌گیرندگان عموماً سعی در تعیین بهینه‌ی اجزای اضافی و تیم‌های تعمیر به منظور رسیدن به مینیمم هزینه و حداکثر دسترس‌پذیری هستند. از آنجا که مساله ازنوع NP-hard می‌باشد، در نتیجه ارائه روشی جهت مدل سازی این نوع مسائل با توجه به انواع محدودیت و دستیابی به حداکثر دسترس‌پذیری امری ضروری در دنیای رقابتی امروز بنظر می‌رسد(Linmin Hu, et al,.2012).

۱-٤. اهداف تحقیق

براساس چارچوب سیستم تعمیرپذیر سری- موازی روش‌های زیادی برای تعیین پارامترهای بهینه‌ی اجزاء از قبیل برنامه‌ریزی پویا[1]، برنامه‌ریزی عدد صحیح[2]، برنامه‌ریزی عدد صحیح غیرخطی[3]، الگوریتم‌های ابتکاری و فرا ابتکاری استفاده شده است. و بطور کلی در چارچوب سیستم سری-موازی پیدا کردن یک راه حل بهینه تحت شرایط محدودیت مختلف بسیار دشوار است (Linmin Hu, et al,.2012).

بطور کلی هدف این مطالعه بصورت زیر می‌باشد:

۱. ارائه یک مدل جهت دستیابی به حداکثر دسترسی پذیری سیستم سری- موازی با توجه به تعمیر پذیر بودن اجزاء و محدویت‌های حجمی، وزنی و هزینه‌ای

۲. بکارگیری الگوریتم رقابت استعماری برای بدست آوردن متغیرهای بهینه‌ی سیستم با زمان کمتر

[1] Dynamic Programming
[2] Integer Programming
[3] Nonlinear Integer Programming

۱-۵. سوالات تحقیق

۱-۵-۱ . سوالات اصلی

مدل مناسب سیستم سری-موازی با اجزای اضافی جهت دستیابی به حداکثر دسترس‌پذیری با توجه به محدودیت‌های حجمی، وزنی و هزینه‌ای چگونه است؟

۱-۵-۲ . سوالات فرعی

آیا الگوریتم رقابت استعماری می‌تواند الگوریتم مناسبی برای حل مسائل NP-hard باشد؟

نقصان روش‌های دیگر حل مانند حل دقیق مسأله چیست؟

۱-٦. فرضیه تحقیق

از آنجایی که مدل‌های ریاضی مکانیزمی در خودشان دارند که خود پایه و اساس مدل سازی و فضای مربوطه آن‌ها را تأیید می‌کند لذا فرضیه‌ای هنگام مدل سازی نمی‌توان در تحقیق متصور شد.

۱-۷. نوع روش تحقیق

طرح حاضر از نظر ماهیت روش، تحلیلی- توصیفی از جنبه کاربردی می‌باشد به خاطر این که به توصیف نظامند ویژگی‌های سیستم‌های سری- موازی با اجزای اضافی پرداخته و پس از مدل سازی دسترس‌پذیری با استفاده از الگوریتم رقابت استعماری به حل مدل پرداخته شده است.

۱-۸. روش گردآوری اطلاعات

به منظور مطالعه ادبیات، سوابق مساله و موضوع تحقیق از مطالعات کتابخانه‌ای استفاده شده است.

۱-۹. ابزار گردآوری اطلاعات

با توجه به این که روش گردآوری اطلاعات کتابخانه‌ای است از متن خوانی و فیش برداری، آمارخوانی و استفاده از جدول و روش‌های ترکیبی در مطالعات کتابخانه‌ای استفاده شده است.

۱-۱۰. تعاریف واژه‌ها و اصطلاحات فنی و تخصصی

۱-۱۰-۱ . قابلیت اطمینان سیستم

قابلیت اطمینان یک سیستم عبارت است از احتمال این که سیستم، کار مورد نظر را تحت شرایط معین در فاصله زمان مشخص بدون خرابی انجام دهد (احتمال انجام کار در زمان مشخص) (شریفی و همکاران،۱۳۹۱).

۱-۱۰-۲ . سیستم تعمیرپذیر

سیستم تعمیرپذیر به سیستمی گفته می‌شود که می‌تواند به منظور اجرای عملیات رضایت بخش به حالت اول برگردانده شود و این برگشت می‌تواند به وسیله هر فعالیتی از جمله جایگزین کردن قطعات، تعمیر آن‌ها یا تغییر تنظیمات صورت گیرد (شریفی و همکاران،۱۳۹۱).

۱-۱۰-۳ . سیستم‌های غیرقابل تعمیر (تعمیرناپذیر)

سیستم غیرقابل تعمیر سیستمی است که اگردر آن عضوی دچار خرابی و نقص گردد، باید از سیستم خارج شده و سیستم با جایگزین کردن آن به فعالیت خود ادامه دهد (شریفی و همکاران،۱۳۹۱).

۱-۱۰-٤ . دسترس‌پذیری

دسترس پذیری عبارت‌است از احتمال این که یک سیستم زمانی که برای انجام یک عمل فراخوان می‌شود، خراب نباشد. به عبارت دیگر دسترس‌پذیری عبارت است از احتمال این که قطعه یا کالا در لحظه t از انجام کار معین تحت شرایط لازم باز نماند (شریفی و همکاران،۱۳۹۱).

۱-۱۰-۵ . سیستم‌های k ازN

فرض کنید سیستمی متشکل از n واحد باشد آنگاه، سیستم را k از n گویند اگر عملکردش مستلزم عملکرد حداقلk واحد از n واحد باشد (k≤n) (Karin S. de Smidt-Destombes, et al.,2007).

۱-۱۰-۶ . فرآیند مارکوف

خاصیت مارکوفی یک فرآیند را می‌توان به زبان ریاضی نیز بیان کرد. مجموعه‌ای از متغیرهای تصادفی $\{x(t),t\geq 0\}$ را در نظر بگیرید. اگر x(t) طبق فرآیند مارکوف عمل کند، به ازای تمام مقادیر $t_1 < t_2 < \ldots < t_n < t_n+1$ رابطه زیر برقرار است (مدرس و همکاران،۱۳۹۱).

$$P\left[x(t_{n+1}) \leq x \middle| x(t_1) = x_1, x(t_2) = x_2, \ldots x(t_n) = x_x\right] = P\left[x(t_{n+1}) \leq x \middle| x(t_n) = x_n\right] \quad (۱-۱)$$

۱-۱۰-۷ . زنجیره‌های مارکوف

زنجیره‌های مارکوف حالت خاصی از فرآیند مارکوف است که در آن هم پارامتر t و هم حالت سیستم، فقط مقادیر گسسته را انتخاب می‌کند. بر این اساس، یک رشته متغیرهای تصادفی $x_1, x_2, \ldots x_n$ را زنجیره مارکوف می‌نامند،اگر به ازای تمام مقادیر n و تمام حالت‌های j, i رابطه زیر برقرار باشد (مدرس و همکاران،۱۳۹۱).

$$P\left[x_{n+1} = j \middle| x_1 = i_1, x_2 = i_2, \ldots x_n = i\right] = P\left[x_{n+1} = j \middle| x_n = i\right] \quad (۲-۱)$$

۱-۱۰-۸ . مسائل Np-hard

مجموعه Np-hard شامل چند هزار مساله‌ی مختلف با کاربردهای فراوان است که تاکنون برای آن‌ها راه حل سریع و قابل انجام در زمان معقول پیدا نشده‌است و به احتمال زیاد در آینده نیز یافت نخواهد شد. این که راه حل سریعی برای آن‌ها وجود ندارد هم اثبات شده‌است. البته ثابت شده است که اگر فقط برای یکی از این مساله‌ها راه حل سریعی پیدا شود، این راه حل موجب حل سریع سایر مساله‌ها خواهد شد. البته احتمال پیدا شدن چنین الگوریتمی ضعیف است. منظور از راه حل سریع آن است که زمان اجرای آن با اندازه‌ی ورودی مساله به صورت چندجمله‌ای رابطه داشته باشد(http://fa.wikipedia.org).

۱-۱۰-۹ . الگوریتم رقابت استعماری

الگوریتم رقابت استعماری روشی در حوزه محاسبات تکاملی است که به یافتن پاسخ بهینه مسائل مختلف بهینه سازی می‌پردازد. این الگوریتم با مدل سازی ریاضی فرایند تکامل اجتماعی – سیاسی، الگوریتمی برای حل مسائل ریاضی بهینه سازی ارائه می‌دهد(آتش پزگرگری، ۱۳۸۷).

فصـــــل دوم

مروری بر مبـــانی و پیشینه

تحـــقیـــق

۲-۱. مقدمه

در سال۱۹۵۲ یک گروه مطالعاتی در قابلیت اطمینان تجهیزات الکترونیک(AGREE)[1] قابلیت اطمینان را در معنا و مفهوم گسترده‌تری ارائه کردند: «قابلیت اطمینان نشان دهنده‌ی احتمال انجام عملکرد یا کارکرد مشخص محصولات و دستیابی موفقیت‌آمیز به اهداف در یک برنامه زمانی تحت یک محیط خاص می‌باشد». بطور کلی قابلیت اطمینان در اولویت بالاتری نسبت به کنترل کیفیت قرار داشته و کیفیت بالا معادل قابلیت اطمینان بالا نمی‌باشد، و حتی ممکن است یک سری اجزای مشخص که فرآیند کنترل کیفیت را در انطباق با مشخصات گذرانده باشد وقتی با سایر اجزاء کار می‌کند منجر به مشکلات شود. با پیشرفت سریع تکنولوژی و افزایش پیچیدگی ساختار سیستم، خرابی هر یک از اجزاء ممکن است به خرابی سیستم و یا آسیب‌های جدی منجر شود. برای مثال سیستم یک سلاح، سیستمی دقیق و پیچیده است که شامل چندین زیر سیستم شامل قطعات و قطعات یدکی می‌باشد. و حتی خرابی تنها یک عنصر به احتمال زیاد تأثیر نامطلوبی بر توانایی عملکرد سیستم سلاح دارد و حتی می‌تواند تهدیدی برای امنیت باشد. دسترس‌پذیری سیستم مفهومی مرتبط با قابلیت اطمینان دارد که به مقیاس اندازه‌گیری قابلیت اطمینان سیستم تعمیرپذیر (سیستمی است که به منظور عملکرد نرمال پس از هر خرابی می‌تواند تعمیر شود از قبیل شبکه‌های کامپیوتری، سیستم‌های تولید، نیروگاه برق و ....) اشاره دارد و نشان دهنده‌ی احتمال عملکرد صحیح سیستم‌های تعمیرپذیر مانند ماشین‌آلات در لحظه‌ی خاص می باشد (Ying shen juang, et al., 2008).

۲-۲. تعاریف، مفاهیم و شاخص‌های قابلیت اطمینان

در این قسمت به تعریف و تشریح مفاهیم کلیدی و شاخص‌های قابلیت اطمینان می‌پردازیم.

مجموعه فعالیت‌های که در امور طراحی، تولید، ساخت و نهایتاً استفاده از کالا در جهت تامین و افزایش میزان اطمینان بر کارکرد آن، انجام می‌گیرد، در بحث مهندسی قابلیت اطمینان قرار دارد. نکته دیگر آن است که قابلیت اطمینان بر معتبر بودن دلالت نمی‌کند؛ قابلیت اطمینان دقت را نشان می‌دهد. روشن است که عموم مهندسان و طراحان باید از مفاهیم اساسی و بنیادی کاربرد روش‌های ارزیابی قابلیت اطمینان آگاه باشند. زیرا قوانین امروز،

1The Advisory Group on The Reliability of Electronic Equipment

تامین کنندگان مواد و لوازم، طراحان و سازندگان محصول را مسئول خسارت‌های وارد بر مصرف‌کنندگان، به سبب بروز معایب محصول می‌دانند(شریفی و همکاران،۱۳۹۱).

قابلیت اطمینان را می‌توان به چندین روش بیان کرد:

– توانایی یک وسیله یا سیستم برای این که عملکردی مطابق طراحی داشته باشد.

– مقاومت یک وسیله یا سیستم در برابر شکست.

– توانایی یک وسیله یا سیستم در اجرای تابع مورد نیازش تحت شرایط معین برای دوره خاصی از زمان

– احتمال این که یک واحد عملیاتی تابع مورد نیازش را در بازه زمانی مشخص تحت شرایط معین اجرا کند.

– توانایی یک سیستم یا اجزاء برای انجام وظایف ضروری تحت یک شرایط مشخص و تعیین شده در یک دوره زمانی معین.

– میزان ثبات کیفیت خدماتی که یک سیستم در حالت طبیعی ارائه می‌کند.

– احتمال انجام کار در زمان مشخص (شریفی و همکاران،۱۳۹۱).

بطور کلی قابلیت اطمینان مفهوم گسترده‌ای را در خود جای داده و مشخصه اصلیش، عملکرد بدون خرابی سیستم[1] در طی انجام ماموریتش می‌باشد و همچنین مشخصاتی از قبیل دسترس‌پذیری، طول عمر، توانایی باز سازی دوباره، ایمنی، ماندگاری و دیگر مشخصه‌های مهم در موضوعات فنی را شامل می‌شود. قابلیت اطمینان به تنهایی هدف نهایی طراحی مهندسی نمی‌باشد یک طراحی می‌تواند کاملاً در شرایط آزمایشگاهی قابل اطمینان بوده ولی در عین حال بیش از اندازه به محیط واقعی حساس باشند و یا برای مثال در طی مدت عملکردش باعث تولید آلودگی غیر قابل قبولین شود که محیط طبیعی را آلوده می‌کند(Ushakov, 2012).

در ادامه در مورد برخی مشخصه‌هایی بحث می‌کنیم که به مفهوم قابلیت اطمینان مرتبط می‌باشند.

۲-۲-۱ . دسترس پذیری

طبق تعریف، بدیهی است که قابلیت اطمینان به عبارتی مبین تداوم عملکرد بدون از کار افتادن می‌باشد.(برای مثال در انجام رساندن یک ماموریت) و لذا قابلیت اطمینان عبارت خواهد بود از احتمال باقی ماندن سیستم و یا یک قطعه در شرایط عملکرد بدون وقوع از کار افتادن. باید دقت شود که این‌گونه تلقی از قابلیت اطمینان مقیاس کاملاً نامناسبی برای سیستم‌ها با عملکرد ممتد و تعمیرپذیر می‌باشد زیرا از کار افتادن در این نوع سیستم‌ها قابل

1 Failure-free

پذیرش نیست (مانند نیروگاه تامین برق). در این حالت مفهوم و مقیاس دیگری بنام دسترس‌پذیری مناسبت می‌یابد، و آن عبارت است از احتمال این که یک سیستم زمانی که برای انجام یک عمل فراخوان می‌شود خراب نباشد، به عبارت دیگر دسترس پذیری عبارت است از احتمال این که قطعه یا کالا در لحظه t از انجام کار معین تحت شرایط لازم باز نماند (شریفی و همکاران،۱۳۹۱). این دو مفهوم(قابلیت اطمینان و دسترس پذیری) در اینجا نشان می‌دهد که نمی‌توان صرفا یک مقیاس فراگیر برای بیان کیفیت عملکرد بکار برد همچنین شاخص های دیگری نیز علاوه بر این دو مطرح است که هر یک جایگاه کاربردی خاص خود را دارد که در ادامه به معرفی برخی از آن‌ها پرداخته خواهد شد.

۲-۲-۲ . **قابلیت نگهداری**[1]

عملکرد بدون خرابی بدون شک یکی از مشخصه‌های مهم در قابلیت اطمینان می‌باشد. اما هر سیستمی بعد از مدتی دچار خرابی شده و در صورت تعمیرپذیر بودن نیازمند بازسازی[2] می‌باشد. در سیستم تعمیرپذیر، قابلیت نگهداری به زمان بازسازی سیستم از زمان شروع تعمیر تا راه‌اندازی دوباره سیستم اشاره دارد. که میانگین زمان برای بازسازی[3] از شاخص‌های آن می‌باشد.

لازم به ذکر است زمان بازسازی شامل زمان آماده‌سازی[4] جهت تعمیر، زمان تعمیر[5] و زمان استارت[6] دوباره سیستم می‌باشد. در نتیجه بازسازی نسبت به تعمیر زمان بیشتری را به خود اختصاص می‌دهد. کیفیت و زمان آن به عواملی همچون کیفیت و تعداد تعمیرکاران یا دسترس بودن ابزار، مواد و ... وابسته است(http://www.ronamax.com).

۲-۲-۳ . **ایمنی**

با بزرگ‌تر شدن صنایع مشکل ایمنی توجه بیشتری را به خود جلب کرده است. واضح است که هدف اصلی یک عملیات فقط عملکرد اصلی آن نیست بلکه یک عملیات موفق عملیاتی است که برای سلامت افراد خطرناک نباشد و به طبیعت آسیبی نرساند یکی از فاجعه‌های صنعتی جهان، تراژدی گاز شهر بویال در سال ۱۹۸٤ بوده که

1 Maintainability
2 Restore
3Mean Time To Restore (MTTR)
4 Restore Time
5 Repair Time
6 Startup Time

در اتحادیه کاربید[1] در محدوده کارخانه آفت‌کش در هند رخ داده است. این فاجعه بزرگ منجر به مرگ آنی حدود هفت هزار نفر شده است. البته حدود هشت هزار نفر دیگر نیز بر اثر بیماری‌های مرتبط با مسمومیت شیمیایی جان خود را از دست داده و حدود نیم میلیون نفر هم دچار مشکلات جدی شدند. سپس در سال ۱۹۸٦ انفجار و آتش‌سوزی در نیروگاه هسته‌ای چرنونیل در اتحادیه جماهیر شوری سابق اتفاق افتاده که بر اثر آن مقدار زیادی از آلودگی رادیواکتیو به داخل جو انتشار یافت و بیشترین انتشارش در اروپا و غرب اتحادیه جماهیر شوری بود. این حادثه به عنوان بزرگ‌ترین حادثه کارخانه نیروگاه اتمی در غرب شناخته می‌شود. هزاران کارگر در آن واحد جان خود را از دست داده و حدود یک میلیون نفر دچار مرگ سرطانی بین سال‌های ۱۹۸٦ و ۲۰۰٤ شدند که نتیجه آلودگی رادیواکتیو بوده است.

۲-۲-٤ . ماندگاری

ماندگاری، یعنی حفظ عملکرد و ایمنی سیستم در مواجه با عوامل طبیعی. در نتیجه قابلیت ماندگاری کاملاً در ارتباط با مسائل قابلیت اطمینان و ایمنی می‌باشد.

در این برهه از زمان مسأله قابلیت ماندگاری سیستم‌های انرژی در مقیاس بزرگ بدلیل وابستگی صنایع به انرژی و قریب‌الوقوع بودن عوامل طبیعی مانند سیل، زلزله بسیار مهم است.

ببرای مثال در ۱۱ مارچ ۲۰۱۱ یک سونامی شدید که بر اثر یکی از بزرگ‌ترین زمین لرزه‌ای که تاکنون ثبت شده(حدود ۹ ریشتر) بوجود آمد و سواحل شرقی ژاپن را درنوردید و باعث خسارت به ۵ راکتور و دو نیروگاه آن کشور شده است که زیان‌های مالی و جانی در حد حادثه‌ی چرنوبیل کشور ژاپن تحمیل کرده است(Ushakov, 2012).

۲-۲-۵ . پایداری[2]

یک طراحی و یا یک سیستم باید توانایی برگشت به حالت عملکرد زمان را بعد از تاثیرات داخلی و خارجی مانند تغییرات درجه حرارت در فصول مختلف را داشته باشد(Ushakov, 2012).

1Union Carbide
[2] Stability

۲-۲-٦ . قابلیت دوام[1]

قابلیت اطمینان در مفهوم خود مشخصه‌ای به عنوان قابلیت دوام را نیز در بردارد. برای مثال برخی از اجزاء سیستم مکانیکی دچار خرابی و یا شکستن می‌شوند که در چندین هزار ساعت اول قابل اطمینان می‌باشند. به هر حال بعد از گذشت دوره زمانی به موجب فرآیند خستگی، خراب می‌شوند و تبدیل به یک جزء غیر قابل قبول برای استفاده می‌شوند(Ushakov, 2012).

۲-۳. مدل سازی شبکه و ارزیابی قابلیت اطمینان انواع سیستم‌ها

۲-۳-۱ . مفاهیم مدل سازی شبکه[2]

قبل از کاربرد شیوهای تحلیل شبکه‌ها برای ارزیابی قابلیت اطمینان ضرورتاً باید درک کاملی از روابط میان سیستم و مدل سازی شبکه حاصل باشد. در مرحله نخست باید توجه شود که در سیستم‌های واقعی ساختار جانمائی عضوهای شبکه الزاماً همانند و یکسان نیستند. به همین جهت یک تحلیل‌گر قابلیت اطمینان باید درک کاملی از عملکردها و الزامات یک سستم داشته باشد تا امکان بررسی‌های کمی برای وی فراهم شود(رضائیان، ۱۳۹۳). در ادامه به تعریف‌هایی در این خصوص خواهم پرداخت.

۲-۳-۱-۱ . سیستم[3]، زیرسیستم[4]، واحد[5]

یک واحد کوچک‌ترین جزء غیر قابل تقسیم در پایین‌ترین سطح از چارچوب تجزیه و تحلیل قابلیت اطمینان می‌باشد و یک سیستم یکی از بالاترین سطح سلسله مراتبی در طراحی برای انجام وظایف در نظر گرفته می‌شود البته مفهوم سیستم و واحد به هم مرتبط هستند. بطوری که یک سیستم در تجزیه و تحلیل‌های مقیاس‌های بزرگ می‌تواند به عنوان واحد در نظر گرفته شودو یا بالعکس.

---

[1] Durability
[2] Network modeling concept
[3] System
[4] Subsystem
[5] Unit

و زیرسیستم می‌تواند یک قسمت از سیستم باشد که به قصد انجام یک وظیفه خاص و یا یک بخش سازنده، مجزا در نظر گرفته شود. و شاخص‌های یک زیر سیستم می‌تواند از طریق شاخص‌های متناظری از واحد و زیر سیستم‌هایش بیان گردد.

۲-۳-۱-۲ . سیستم‌های تعمیرپذیر[1]

سیستم تعمیرپذیر به سیستمی گفته می‌شود که می‌تواند به منظور اجرای عملیات رضایت بخش به حالت اول برگردانده شود و این برگشت می‌تواند به وسیله هر فعالیتی از جمله جایگزین کردن قطعات، تعمیر آن‌ها یا تغییر تنظیمات صورت گیرد. به عبارت دیگر تعمیرپذیری عبارت است از احتمال این که تعمیر قطعه یا کالای خراب شده در زمان $T_0=0$ تحت شرایط معینی در فاصله $[T_0, T]$ انجام گیرد(شریفی و همکاران،۱۳۹۱).

هنگامی که از نرخ خرابی[2] صحبت می‌شود سیستم‌های تعمیرپذیر مدنظر است. در حالی که وقتی این نرخ روی سیستم‌های غیر قابل تعمیر مطرح شود. تنها به اولین زمان خرابی اشاره می‌کند. و منظور از خرابی یا شکست[3] متوقف شدن توانایی نهاد[4] یا سیستم برای انجام کار معین تحت شرایط لازم می‌باشد.

۲-۳-۱-۳ . سیستم‌های تعمیر ناپذیر[5]

سیستم تعمیرناپذیر سیستمی است که اگر در آن نهاد یا واحدی دچار خرابی و نقص گردد، باید از سیستم خارج شده و سیستم با جایگزین کردن آن به فعالیت خود ادامه دهد. موشک‌ها و ماهواره‌برها، نمونه‌هایی از سیستم‌های تعمیر ناپذیر می‌باشد(شریفی و همکاران،۱۳۹۱).

۲-۳-۲ . ریاضیات آنالیز قابلیت اطمینان

فرض کنید T متغیر تصادفی نامنفی پیوسته‌ای است که عمر مفید (طول عمر یا زمان پیش از خرابی) یک سیستم (جزء، وسیله، قطعه و ...) را نشان می‌دهد. قابلیت اطمینان یا پایائی که آن را با $R(t)$ نمایش می‌دهیم، احتمال آن

---

[1] Repairable systems
[2] Hazard rate
[3] Failure
[4] Entity
[5] Non repairable

است که یک سیستم (جزء، وسیله، قطعه و...) بیش از دورهٔ زمانی t فعـال و سـالم بـاقی بمانـد. در برخـی مـوارد قابلیت اطمینان را به «تابع بقا» نیز می‌شناسند(شریفی و همکاران،۱۳۹۱).

به زبان ریاضی، فرض کنید که t دوره زمانی مأموریت و T مدت زمان عملکرد صحیح سیستم (محصول، جزء، وسیله و ...) باشد. آن‌گاه قابلیت اطمینان می‌تواند بصورت زیر عنوان گروه:

$$R(t) = p_r(T \succ t) = 1 - F(t) = \int_t^{\infty} f(x)dx \qquad (۱-۲)$$
$$\lim_{t \to \infty} R(t) = 0, R(0) = 1, R(t) \succ 0$$

بنابراین، احتمال خراب شدن قطعه‌ای در بازه زمانی t_1 تا t_2 عبارت است از:

$$P(t_1 < T < t_2) = \int_{t_1}^{t_2} f(t)\,dt = F(t_2) - F(t_1) \qquad (۲-۲)$$

عدم اطمینان در نقطه متقابل قابلیت اطمینان قرار دارد و برابر است با احتمال عدم موفقیت محصول برای حفظ کارایی‌اش در طول مأموریت می‌باشد و به صورت رابطه زیر تعریف می‌شود (http://ssd2012aut.ac.ir):

$$1\text{-}R(t)=P_r\,(T< t) = F(t) \qquad (۳-۲)$$

توجه داشته باشید که R(t) احتمال آن است که زمان خرابی بزرگتر یا مساوی t باشد. حال آن‌کـه F(t) احتمـال آنست که خرابی قبل از زمـان t رخ دهـد. R(t) را تـابع قابیـت اطمینـان و F(t) را تـابع توزیـع تجمعـی[1] (CDF) می‌نامند.

۲-۳-۲-۱ . نرخ خرابی یا نرخ شکست

نرخ شکست عبارتست از احتمال این که محصولی در بازه زمانی [t+Δt] دچار خرابی شود در عین حال تا قبل از زمان t دارای عملکرد صحیح بوده باشد. تعریف فوق را می‌توان بصورت روابط ریاضیاتی زیر بیان کرد:

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{p(t < T < t + \Delta t \mid T > t)}{\Delta t}$$
$$= \lim_{\Delta t \to 0} \frac{-R(t + \Delta t) + R(t)}{\Delta t * R(t)} = \frac{\frac{-dR}{dt}}{R(t)} \qquad (٤-۲)$$

با حل معادله‌ی (۲-٤) خواهیم داشت:

[1] Cumulative Distribution

$$R(t) = e^{-\int_0^t \lambda(t)dt} \qquad (۵-۲)$$

از طرفی می‌دانیم که تابع چگالی احتمال از مشتق‌گیری تابع توزیعی تجمعی بدست می‌آید:

$$f(t) = \frac{dF}{dt} = -\frac{dR}{dt} \qquad (۶-۲)$$

با استفاده از رابطه‌ی (۲-۵) نتیجه می‌گیریم:

$$\lambda(t) = \frac{f(t)}{R(t)} \qquad (۷-۲)$$

نرخ خرابی یک پارامتر مهم در آنالیز قابلیت اطمینان است که اغلب تابعی از زمان می‌باشد، و برخی از قطعات مانند قطعات الکترونیکی دارای نرخ خرابی عمدتاً ثابت می‌باشند یعنی از توزیع نمایی پیروی می‌کنند(http://ssd2012aut.ac.ir).

در مورد توزیع نمایی، با توجه به بی‌حافظه بودن توزیع، نرخ خرابی یک مقدار ثابت نسبت به زمان خواهد بود. به این معنی که با گذشت زمان انتظار نداریم تغییری در تمایل سیستم به ایجاد خطا به وجود آید. برای توزیع‌های دیگر مانند توزیع وایبول، داگ نرمال، تابع نرخ خرابی نسبت به زمان، ثابت نیست (شریفی و همکاران،۱۳۹۱).

با توجه به توضیحات ذکر شده خواهیم داشت:

$$\lambda(t) = \lambda = cons\tan t \Rightarrow R(t) = e^{-\int_0^t \lambda(t)dt} = e^{-\lambda t} \qquad (۸-۲)$$

$$f(t) = -\frac{dR}{dt} = \lambda e^{-\lambda t} \qquad (۹-۲)$$

۲-۳-۲-۲ . **میانگین زمان خرابی**[1]

در تحلیل قابلیت اطمینان، میانگین زمان منتهی به خرابی (MTTF) اغلب مورد استفاده است که به صورت زیر محاسبه می‌شود(شریفی و همکاران،۱۳۹۱).

$$MTTF = E(t) = \int_0^\infty t\ f(t)dt = \int_0^\infty R(t)dt \qquad (۱۰-۲)$$

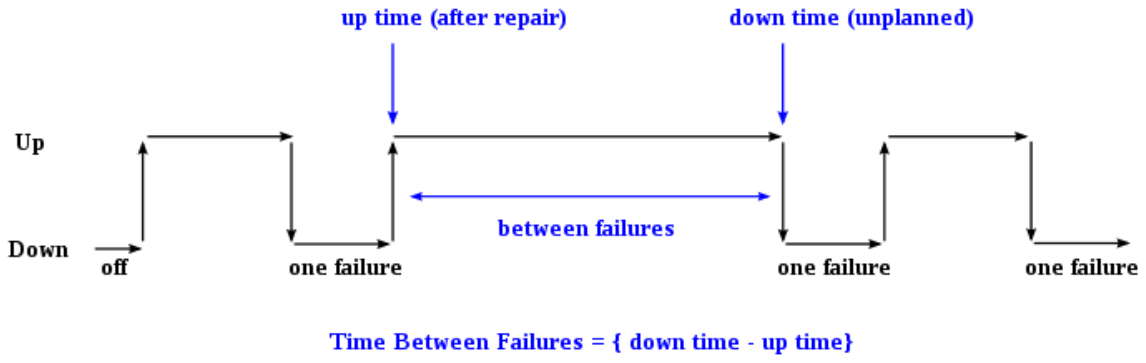
[1] Mean Time to Failure (MTTF)

۲-۳-۲-۳ . میانگین زمان بین دو خرابی[1]

یک راه برای بیان وقوع شکست، استفاده از متوسط زمان بین شکست‌های پی‌درپی است که اغلب به MTBF شناخته می‌شود. اختلاف اساسی بین MTBF و MTTF (متوسط زمان شکست) در موارد استفاده از آن‌ها است. شاخص MTBF یک شاخص کلیدی قابلیت اطمینان برای سیستمی است که قابلیت تعمیر شدن داشته باشد در حالی که MTTF زمان مورد انتظار برای اولین خرابی یا شکست سیستم غیر قابل تعمیر است. بنابراین در عمل،MTBF مجموع زمان یک سیکل خرابی و یک سیکل در حال کار است.

$$MTBF=MTTF+MTTR(MDT) \qquad (۱۱-۲)$$

در رابطه (۲-۱۱) MTTR بیان‌گر میانگین زمان تعمیر[2] یک سیستم بوده و همچنین معادل میانگین زمان خرابی[3] می‌باشد.



شکل ۲-۱ (شریفی و همکاران،۱۳۹۱)

طبق نمودار شکل ۲-۱ می‌توان میانگین زمان بین دو خرابی را بصورت زیر تعریف کرد:

$$MTBF = \frac{\sum(Downtime - Uptime)}{Number\ of\ Failures} \qquad (۱۲-۲)$$

[1] Mean Time Between Failure (MTBF)
[2] Mean Time To Repair
[3] Mean Down Tim (MDT)

۲-۳-۳ . انواع سیستم‌ها در ارزیابی قابلیت اطمینان

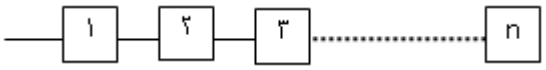
۲-۳-۳-۱ . سیستم‌های با شبکه سری (متوالی)[1]

یک سیستم سری سیستمی است که خرابی هر یک از واحدها باعث از کار افتادن کل سیستم می‌شود، و این نوع از سیستم‌ها بوسیله واحدها و یا زیر سیستم‌هایش به یک دیگر متصل می‌شوند. و ساختار سری یکی از ساختارهای رایج در طراحی‌های مهندسی بوده و دارای کاربرد بسیار می‌باشد(Ushakov, 2012).

در مهندسی قابلیت اطمینان، برای توصیف ارتباط منطقی بین واحدهای سیستم، از نمودارهای بلوک دیاگرام[2] استفاده می‌شود.

بلوک دیاگرام، نمایشی از چیدمان فیزیکی اجزای یک سیستم است. برای این منظور یک سیستم به زیر سیستم‌های نرم‌افزاری و سخت افزاری تقسیم می‌گردد و سپس نمایش‌های بلوکی درون هر یک از این زیر سیستم‌ها ایجاد می‌شود. این بلوک‌ها به صورت سری یا موازی و یا پیچیده به یک دیگر رشته شده‌اند تا نشان دهند که چگونه هر جزء باید برای موفقیت در سیستم عمل کند. با توجه به نمودار بلوکی و تکنیک‌های متناسب با هر چیدمان می‌توان قابلیت اطمینان یک سیستم را محاسبه نمود(شریفی و همکاران،۱۳۹۱).

برای یک سیستمی که از n واحد تشکیل شده باشد و بصورت سری با یک دیگر در ارتباط باشند بلوک دیاگرام (RBD) آن بصورت شکل ۲-۲ (بلوک دیاگرام سیستم سری با n عضو متوالی) خواهد بود.



شکل ۲-۲ (بلوک دیاگرام سیستم سری با n عضو متوالی(رضائیان، ۱۳۹۳))

که در آن قابلیت اطمینان (احتمال عدم خرابی- پایایی) واحد k ام توسط $R_k(t)=Pr\{T_k>t\}$ نشان داده می‌شود.

در مهندسی قابلیت اطمینان، $q(t)$ به عنوان احتمال از کار افتادن و یا عدم اطمینان شناخته می‌شود که رابطه‌ی زیر را با قابلیت اطمینان دارد:

$$q(t)=1-R(t) \qquad (۱۳-۲)$$

[1] Series system
[2] Reliability Block Diagrams (RBD)

اگر واحدهای سیستم بسیار قابل اطمینان باشند یعنی $Max\ q_k(t) \prec\prec \frac{1}{n}$ $1 \le k \le n$ باشد آنگاه رابطه‌ی (۲-۱٤) برقرار می‌باشد.

$$R(t) = \prod_{k=1}^{n}[1 - q_k(t)] \approx 1 - \sum_{i=1}^{n} q_k(t) \qquad (۲-۱٤)$$

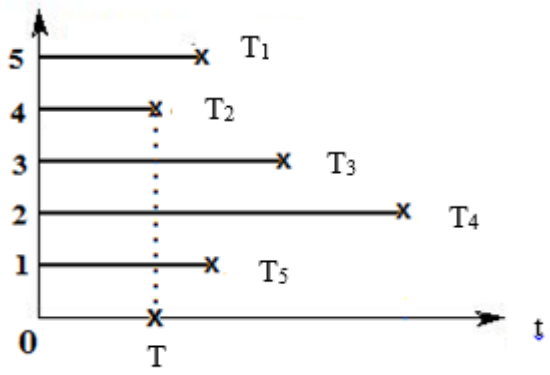
از رابطه (۲-۱٤) می‌توان به نتایج زیر رسید(Ushakov, 2012):

- قابلیت اطمینان یک سیستم سری افزایش (کاهش) می‌یابد اگر قابلیت اطمینان تک‌تک واحدها افزایش (کاهش) یابد.
- قابلیت اطمینان یک سیستم سری کاهش (افزایش) می‌یابد اگر تعداد واحدهای سیستم افزایش (کاهش) یابد.
- قابلیت اطمینان سیستم از قابلیت اطمینان تک‌تک واحدها کمتر می‌باشد.

اگر زمان تصادفی تا شکست[1] سیستم را با T و زمان تصادفی تا شکست واحدها را با T_k نشان دهیم بطوری که $1 \le K \le n$ باشد، زمان تصادفی تا شکست سیستم سری بصورت رابطه (۲-۱۵) می‌باشد (Ushakov, 2012):

$$T = \min_{1 \le K \le n} T_k \qquad (۲-۱۵)$$

شکل ۲-۳ (نمایش TTF یک سیستم سری) نیز بیانگر رابطه‌ی(۲-۱۵) می‌باشد.



شکل ۲-۳ (نمایش TTF یک سیستم سری(Ushakov, 2012))

اغلب از عبارت‌های بولین[2] برای توضیح ساختار قابلیت اطمینان سیستم استفاده می‌شود. برای یک ساختار سری عبارت بولین بصورت زیر می‌باشد:

[1] Time To Failure (TTF)
[2] Boolean

$$\Phi(x) = \bigcap_{i=1}^{n} x_i \qquad (۱۶-۲)$$

که $X = (x_1 , x_2 , ..., x_n)$ می‌باشد. برای واحدهای مستقل $R(t)=E\{\phi(X(t))\}$، و با استفاده از قضیه ضرب‌ها می‌توانیم بنویسیم:

$$R(t) = pr\left\{\bigcap_{i=1}^{n} x_i = 1\right\} = \Pr\{T_1 > t\} \cdot \Pr\{T_2 > t\} ... \{Tn \geq t\} = \qquad (۱۷-۲)$$

$$= \prod_{k=1}^{n} \Pr\{T_k \geq t\} = \prod_{k=1}^{n} R_k(t)$$

اگر هر جزء دارای طول عمر با توزیع نمایی با نرخ ثابت λ_i باشد، قابلیت اطمینان سیستم از طریق رابطه‌ی زیر بدست می‌آید(شریفی و همکاران،۱۳۹۱):

$$R_s(t) = \prod_{k=1}^{n} R_k(t) = \prod_{k=1}^{n} \exp(-\lambda_k t) = \exp\left[-t\sum_{k=1}^{n} \lambda_k\right] \qquad (۱۸-۲)$$

برای سیستمی که واحدهای آن بسیار قابل اعتماد باشند، $Max\ \lambda_k << \frac{1}{n}$ $1 \leq k \leq n$ می‌توان از تقریب زیر استفاده کرد(Ushakov, 2012):

$$p(t) = 1 - t\sum_{k=1}^{n} \lambda_k \qquad (۱۹-۲)$$

اگر تمام واحدهای سیستم همانند و یکسان باشند، رابطه‌ی(۲-۲۰) برقرار می‌شود(Ushakov, 2012):

$$p(t) = \exp(-\lambda nt) \qquad (۲۰-۲)$$

برای درک بهتر یک سیستمی را در نظر بگیرید که اجزای آن دارای R(t)=0.999 باشد در جدول ۲-۱(رابطه قابلیت اطمینان سیستم سری با تعداد واحدهای آن) نشان داده می‌شود که قابلیت اطمینان سیستم با افزایش تعداد واحدها کاهش می‌یابد.

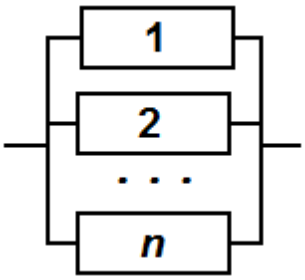
جدول ۲-۱(رابطه قابلیت اطمینان سیستم سری با تعداد واحدهای آن)

n	۱۰	۱۰۰	۱۰۰۰	۱۰۰۰۰
R(t)	0.99005	0.904837	0.367879	تقریباً صفر

۲-۳-۳-۲ . سیستم‌های با شبکه موازی[1]

دو یا چند جزء که موازی هم یا به عبارتی عضو مازاد یک دیگر باشند، سیستم موازی نامیده می‌شوند. در این پیکربندی زمانی توقف و خرابی کامل سیستم پیش خواهد آمد که تمام اجزای سیستم خراب شوند. به عبارتی، اگر یک یا تعداد بیشتری از واحدها کار کنند، سیستم همچنان به عملکرد خود ادامه می‌دهد(شریفی و همکاران،۱۳۹۱).

بلوک دیاگرام سیستم موازی با n واحد در شکل زیر نشان داده شده است.

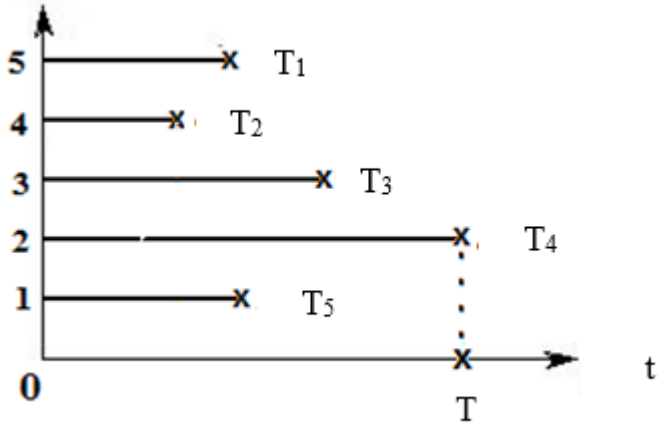


شکل ۲-۴ (بلوک دیاگرام سیستم موازی با n عضو موازی(Ushakov, 2012))

اگر زمان تصادفی تا شکست *(TTF)* سیستم را با T و *TTF* واحدها را با T_k نشان دهیم، بطوری که $1 \le k \le n$ باشد، آن‌گاه از تعریف سیستم موازی خواهیم داشت (Ushakov, 2012):

$$T = \underset{1 \le k \le n}{Max}\ T_k \qquad (۲۱-۲)$$

حالت ذکر شده بسادگی از شکل ۲-۵ (نمایش TTF یک سیستم موازی) قابل فهم می‌باشد.



شکل ۲-۵ (نمایش TTF یک سیستم موازی(Ushakov, 2012))

[1] Parallel system

با توجه به تعریف ذکر شده برای یک سیستم موازی می‌توان آن را بصورت تابع بولین[1] توصیف کرد:

$$\phi(x) = \bigcup_{k=1}^{n} x_k = x_1 \cup x_2 \cup ... \cup x_n \qquad (۲۲-۲)$$

عبارت (۲-۲۲) را می‌توان با استفاده از قوانین جبری دمورگان[2] تغییر شکل داد که در آن $\overline{X}$ متمم X می‌باشد.

$$x_1 \cup x_2 = \overline{\overline{x}_1 \cap \overline{x}_2} \qquad (۲۳-۲)$$

$$\hat{\bigcup}_{k=1} x_k = \overline{\hat{\bigcap}_{k=1} \overline{x}_k} \qquad (۲۴-۲)$$

پیرو رابطه‌ی (۲-۲۴) برای قابلیت اطمینان خواهیم داشت:

$$R(t) = E\{\phi(x)\} = 1 - \prod_{i=1}^{n} q_i \qquad (۲۵-۲)$$

در رابطه (۲-۲۵)، $P_r\{x_k = 0\} = q_k$ می‌باشد. همچنین با استفاده از تعریف سیستم موازی و استفاده از تئوری ضرب خواهیم داشت:

$$Q(t) = pr\{(T_1 < t) \cap (T_2 < t) \cap .... \cap (T_n < t) = \prod_{k=1}^{n} pr\{T_k < t\} = \prod_{k=1}^{n} q_k(t) \qquad (۲۶-۲)$$

در رابطه‌ی (۲-۲۶)، Q(t) احتمال شکست سیستم موازی، Q(t)=1-R(t) و $q_k(t)$ احتمال شکست واحد k، $q_k(t)=1-R_k(t)$ می‌باشد. بنابراین قابلیت اطمینان یک سیستم موازی بصورت زیر تعریف می‌شود:

$$R(t) = 1 - Q(t) = 1 - \prod_{k=1}^{n} q_k(t) \qquad (۲۷-۲)$$

رابطه‌ی(۲-۲۷) بصورت رابطه‌ی (۲-۲۸) نیز قابل استفاده است(Ushakov, 2012).

$$\begin{aligned} R(t) &= R_1(t) + q_1(t)R_2(t) + q_1(t)q_2(t)R_3(t) + ... + q_1(t)q_2(t)...q_{m-1}(t) \cdot R_m(t) \\ &= R_1(t) + q_1(t)\big[R_2(t) + q_2(t)\big[R_3(t) + ... + q_{m-1}(t) \cdot R_m(t)\big]\big] \end{aligned} \qquad (۲۸-۲)$$

رابطه (۲-۲۸) بصورت زیر قابل توضیح می‌باشد :

احتمال عملکرد موفق یک سیستم موازی برابر است با احتمال این که:

---

[1] Boolean Function

[2] Morgan

- اولین واحد در طی زمان t موفق عمل کند.
- یا اگر اولین واحد خراب شود، دومین واحد تا زمان t موفق عمل کند.
- و یا اگر اولین و دومین واحد خراب شود، سومین واحد تا زمان t موفق عمل کند.

.

.

.

- و یا اگراولین تا (m-1) اُمین واحد خراب شود m اُمین واحد تا زمان t موفق عمل کند.

از رابطه‌ی (۲-۲۵) می‌توان نتایج زیر را گرفت:

- قابلیت اطمینان یک سیستم موازی کاهش (افزایش) می‌یابد اگر قابلیت اطمینان هر یک از اجزاء کاهش (افزایش) یابد.
- قابلیت اطمینان سیستم موازی کاهش (افزایش) می‌یابد اگر تعداد واحدها کاهش (افزایش) یابد.
- قابلیت اطمینان سستم موازی از قابلیت اطمینان هر یک از واحدهای خود بیشتر می‌باشد.

در صورتی که در یک سیستم موازی، طول عمر اجزاء دارای توزیع نمایی باشند $(R_k(t) = e^{\lambda kt})$ داریم:

$$R(t) = 1 - \prod_{k=1}^{n} (1 - e^{-\lambda kt}) \qquad (۲۹-۲)$$

برای درک بهتر تأثیر تعداد واحدها بر روی قابلیت اطمینان، یک سیستم موازی با n جزء همسان با R(t)=0.9 را در نظر بگیرید. در جدول ۲-۲ (رابطه قابلیت اطمینان سیستم موازی با تعداد واحدهای آن) می‌توان دید که قابلیت اطمینان بطور قابل توجه‌ای با افزایش تعداد واحدها افزایش می‌یابد.

جدول ۲-۲ (رابطه قابلیت اطمینان سیستم موازی با تعداد واحدهای آن)

n	2	3	4	5	.....	۱۰
R(t)	0/991	0/9991	0/99992	0/999992	....	تقریبا ۱

۲-۳-۳-۳ . سیستم‌های با شبکه k از n

در مهندسی قابلیت اطمینان بکارگیری تکنیک‌های اجزای اضافی برای بهبود قابلیت اطمینان و دسترس‌پذیری سیستم امری رایج می‌باشد. یکی از فرم‌های رایج اجزای اضافی ساختار k-out-of - n:F و k-out- of- n:G می‌باشد که توسط Birn و همکاران در سال ۱۹۶۱ معرفی گردید. سیستم k-out – of- n:G شامل n جزء می‌باشد و کار می‌کند اگر و فقط اگر حداقل k تا از n جزء کار کند. و همچنین k – out – of- n:F نیز شامل n جزء می‌باشد و کار نمی‌کند اگر و فقط اگر حداقل k تا از n جزء کار نکند. بنابراین یک سیستم k-out-of-n:G با یک سیستم (n-k+1)-out-of-n:F برابر است. در ادبیات تحقیق عبارت 'k-out-of-n' برای هر دو حالت k-out-of-n:G و k-out-of-n:F استفاده می‌شود. ولی در عمل اغلب مهندسین از عبارت 'k-out-of n' برای نشان دادن سیستم k-out-of-n:G استفاده می‌کند و حرف G و F را در نظر نمی‌گیرند(Suprasad.et al.,2008).

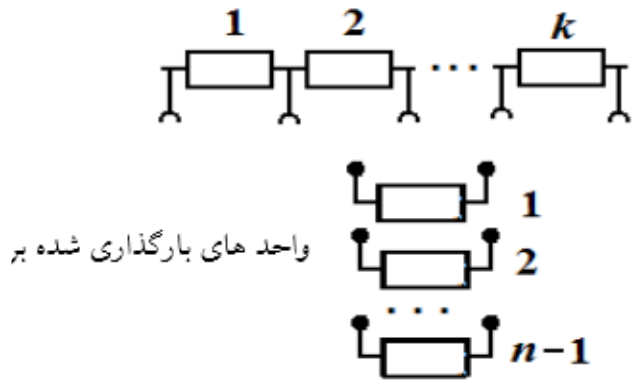
ساختار اجزای اضافی k-out-of-n:G کاربردهای زیادی در سیستم‌های صنعتی و نظامی پیدا کرده است. برای مثال کابل‌های بکار رفته در پل‌ها، سیستم پردازش اطلاعات با نمایشگر ویدوئی چندگانه، سستم‌های ارتباطات با فرستنده‌های چندگانه و سیستم موتور هواپیما مثلاً هواپیمای ٤ موتوره‌ای را در نظر بگیرد که برای پروازش حداقل تنها به ۲ موتور نیاز دارد، و دو موتور آن می‌تواند مازاد محسوب شود. در واقع سیستم موتور این هواپیما یک سیستم 2-out-of-4 می‌باشد.

لازم بذکر است در میان کاربردهای سیستم k-out-of-n، طراحی مدارهای الکترونیکی مانند مدارات مجتمع با مقیاس بسیار بزرگ و تعمیر اتوماتیک نقص‌ها در یک سیستم آنلاین چشمگیرتر می‌باشد(Suprasad.et al.,2008).

بطور خلاصه یک سیستم با این ساختار شامل n واحد می‌باشد و در حالت کارکرد می‌باشد تا زمانی که تعداد واحدهای معیوب به n-k+1 برسد. تابع ساختاری این سیستم‌ها می‌تواند بصورت زیر نوشته شود(Ushakov, 2012).

$$\phi(x)\begin{cases}1 & if \quad \sum_{i=1}^{n} X_i \geq k \\ 0 & ptherwise\end{cases} \qquad (۳۰-۲)$$

در حقیقت این چنین سیستمی را می‌توان به عنوان یک سیستم سری k واحد به همراه n-k واحد اضافی در نظر گرفت. که هر یک از این واحد اضافی می‌تواند جایگزین هر یک از اجزاء خراب شوند. ساختار RBD این نوع از سیستم در شکل زیر نمایش داده شده است.



شکل ۲-۶ (RBD در ساختار K از N برای سیستم تعمیر ناپذیر(Ushakov, 2012))

۲-۳-۳-۳-۱ . سیستم k از n با توزیع طول عمر مستقل و همسان و تعمیر ناپذیر

در یک سیستم k از n با توزیع طول عمر مستقل و یکسان[1]، تعداد اجزاء در حال کار از توزیع دو جمله‌ای با پارامترهای n , p (قابلیت اطمینان هر جزء زمانی که تمامی اجزاء دارای توزیع طول عمر مستقل و همسان هستند)پیروی می‌کنند(یحیی‌تبار، ۱۳۹۱):

$$\text{pr (دقیقاً I جزء کار کند)} = \binom{n}{i} p^i q^{n-i} \quad i = 0, 1, 2, \ldots, n \qquad (۳۱-۲)$$

قابلیت اطمینان سیستم برابر است با احتمال این که تعداد اجزاء در حال کار حداقل k تا باشد (یحیی‌تبار، ۱۳۹۱):

$$R_{k-out-of-n}(t) = \sum_{j=k}^{n} \binom{n}{j} [p(t)]^j [q(t)]^{n-j} = 1 - \sum_{j=0}^{k-1} \binom{n}{j} [p(t)]^{n-j} [q(t)]^j \qquad (۳۲-۲)$$

برای واحدهایی با قابلیت اطمینان بالا (وقتی که $q << \frac{1}{n}$)، می‌توان از تقریب زیرا استفاده کرد (Ushakov, 2012):

$$R_{k-out-of-n} \approx 1 - \binom{n}{k-1} [q(t)]^{n-k+1} \qquad (۳۳-۲)$$

اگر تابع توزیع طول عمر واحدها نمایی باشند می‌توان از رابطه‌ی زیر استفاده کرد (Ushakov, 2012):

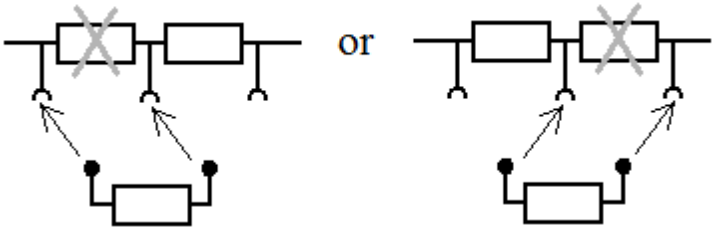
$$R_{k-out-of-n} = \frac{1}{\lambda} \sum_{i=k}^{n} \frac{1}{i} \qquad (۳۴-۲)$$

[1] independent identical distribution (i.i.d)

توجه داشته باشید که وقتی k=n باشد ساختار k از n به یک سیستم سری متداول با n واحد تبدیل می‌شود و همچنین وقتی که k=1 باشد ساختار k از n به یک سیستم موازی متداول با n واحد تبدیل می‌شود. یک سیستم ۲ از ۳ (2-out-of-3) را در نظر بگیرید. برای این سیستم عبارتی بولین به صورت زیر تعریف می‌شود (Ushakov, 2012):

$$\phi(x) = (x_1 \cap x_2 \cap x_3) \cup (\bar{x}_1 \cap x_2 \cap x_3) \cup (x_1 \cap \bar{x}_2 \cap x_3) \cup (x_1 \cap x_2 \cap \bar{x}_3) \quad (۳۵-۲)$$

RBD برای چنین سیستمی در شکل زیر نشان داده شده است.



شکل ۲-۷ (ارتباط یک واحد اضافی بجای واحد از کار افتاده در یک سیستم 2-out-of-3 (Ushakov, 2012))

از رابطه‌ی (۲-۳۲) برای واحدهای همسان خواهیم داشت:

$$R_{2-out-of-3}(t) = E\{\phi(x)\} = P^3 + 3p^2q \quad (۳۶-۲)$$

۲-۳-۳-۳-۲ . سیستم k از N با توزیع طول عمر مستقل، ناهمسان و تعمیر ناپذیر

برای سیستم‌هایی که توزیع طول عمر آن‌ها ضرورتا یکسان نیست می‌توان از روش مجموع مسیرهای مینیمم[1] برای ارزیابی قابلیت اطمینان آن‌ها استفاده کرد.قابلیت اطمینان هر سیستم برابر است با احتمال این که حداقل یک مجموعه مسیر مینیمم در حال کار کردن باشد و عدم قابلیت اطمینان سیستم برابر است با احتمال این که حداقل یک برش مینیمم منجر به شکست شود برای این که یک مجموعه مسیر مینیمم در حال کار باشد، هر جز در مجموعه می‌باید در حال کار باشد. برای این که یک مجموعه برش مینیمم منجر به شکست شود، تمامی اجزاء در مجموعه می‌باید دچار شکست شوند. در یک سیستم k از n، $\binom{n}{k}$ مجموعه مسیر مینیمم و $\binom{n}{n-k+1}$ مجموعه برش مینیمم وجود دارد. هر مجموعه مسیر مینیمم دارای دقیقاً k جزء متفاوت و هر مجموعه برش مینیمم دارای

[1] Minimal path set

دقیقاً n-k+1 جزء است. بنابراین کل مجموعه مسیر مینیمم و مجموعه برش مینیمم مشخص هستند. سوالی که در اینجا باقی می‌ماند این است که احتمال این که حداقل یک مجموعه مسیر مینیمم شامل کلیه اجزاء در حال کار باشد یا احتمال این که حداقل یک مجموعه برش مینیمم شامل کلیه اجزاء خراب باشد چقدر است. هیدمن با استفاده از این شیوه فرمول زیر را برای این سیستم بدست آورد(یحیی‌تبار، ۱۳۹۱):

$$R = \sum_{i=k}^{n} (-1)^{i-k} \binom{i-1}{k-1} \sum_{f_1<f_2<\ldots<f_i} \prod_{f=1}^{i} p_{ft} \qquad (۳۷-۲)$$

۲-۳-۳-۳-۳ . ارزیابی تصادفی سیستم k از N با اجزای دارای طول عمر مستقل و همسان

در بخش‌های قبلی قابلیت اطمینان به عنوان تابعی از زمان بیان نشده بود اما در حقیقت قابلیت اطمینان و دسترس‌پذیری به عنوان تابعی از زمان بیان می‌شوند. در این قسمت ما یک ارزیابی تصادفی از سیستم‌های k-out-of-n را خواهیم داشت.

زمانی که اجزاء در یک سیستم k از n دارای طول عمر مستقل و همسان هستند، تابع پایایی یا قابلیت اطمینان سیستم می‌تواند بصورت زیر نشان داده شود.

$$R_s(t) = \sum_{i=k}^{n} \binom{n}{i} R(t)^i F(t)^{n-i} \qquad (۳۸-۲)$$

برای تابع احتمال تجمعی طول عمر سیستم خواهیم داشت:

$$F_s(t) = 1 - R_s(t) = \sum_{i=0}^{k-1} \binom{n}{i} R(t)^i F(t)^{n-i} \qquad (۳۹-۲)$$

برای تابع چگالی آن هم خواهیم داشت:

$$f_s(t) = \frac{dF_s(t)}{dt} = k \binom{n}{k} f(t) R(t)^{k-1} F(t)^{n-k} \qquad (۴۰-۲)$$

در واقع وقتی سیستم در حال کارکردن است. اجزاء یکی پس از دیگری خراب می‌شوند و در نهایت زمانی که تعداد اجزاء خراب سیستم به N-k+1 برسد سیستم از کار می‌افتد. اگر ما از t_i برای نشان دادن طول عمر جزء i استفاده کنیم، طول عمر سیستم با N-K+1 کوچک‌ترین t_i برابر است(یحیی‌تبار، ۱۳۹۱).

زمانی که اجزاء از توزیع طول عمر نمایی با تابع احتمال تجمعی $F(t)=1-e^{-\lambda t}$ پیروی کنند قابلیت اطمینان و عدم قابلیت اطمینان سیستم بصورت زیر خواهد بود:

$$R_s(t) = \sum_{i=k}^{n} \binom{n}{i} (e^{-\lambda t})(1-e^{-\lambda t})^{n-i} \qquad (۴۱-۲)$$

$$F_s(t) = \sum_{i=0}^{k-1} \binom{n}{i} (e^{-\lambda t})(1-e^{-\lambda t})^{n-i} \qquad (۴۲-۲)$$

۲-۳-۳-۳-٤ . ارزیابی تصادفی سیستم k از N با اجزای دارای طول عمر مستقل و ناهمسان

بطور کلی نوشتن یک رابطه‌ی مشخص برای سیستم k از n زمانی که اجزاء سیستم دارای توزیع طول عمر مستقل و ناهمسانی هستند مشکل است. مانند اجزای یک سیستم با توزیع‌های طول عمر نمایی که جزء iام دارای یک نرخ خرابی ثابت (λ_i (1≤i≤n) می‌باشد. برای مثال قابلیت اطمینان و MTTF یک سیستم 2-out-of-3 بصورت زیر خواهد بود(یحیی‌تبار، ۱۳۹۱):

$$R_s(2-out-of-3) = e^{-(\lambda_1+\lambda_2)t} + e^{-(\lambda_1+\lambda_3)t} + e^{-(\lambda_2+\lambda_3)t} + e^{-(\lambda_1+\lambda_2+\lambda_3)t} \qquad (۴۳-۲)$$

$$MTTF(2,) = \frac{1}{\lambda_1+\lambda_2} + \frac{1}{\lambda_1+\lambda_3} + \frac{1}{\lambda_2+\lambda_3} + \frac{1}{\lambda_1+\lambda_2+\lambda_3} \qquad (۴۴-۲)$$

۲-۳-۳-۳-۵ . سیستم‌های k-out-of-n تعمیرپذیر

در قسمت‌های گذشته در مورد سیستم‌های تعمیرناپذیر توضیح داده شد. در این قسمت به سیستم‌های قابل تعمیر می‌پردازیم. در مورد سیستم‌های تعمیرپذیر مفهومی مهم و ضابطه‌ای برای ارزیابی سیستم استفاده می‌شود و آن دسترسی‌پذیری سیستم است و بنا به تعریف عبارت است از احتمال این که سیستم صرفنظر از پیشینه خرابی‌ها و تعمیرهایی که در گذشته داشته است در زمان t کار کند. و در ادبیات آن‌را با A(t) نشان می‌دهند.

زمانی که یک سیستم k از n شروع به کار می‌کند، تمامی n جزء در وضعیت سالم قرار دارند. هرچه از نظر زمانی به جلو می‌رویم اجزاء یکی پس از دیگری شروع به خراب شدن می‌کنند. سیستم زمانی از کار می‌افتد که تعداد اجزاء در حال کار کمتر از k جز شود یا تعداد اجزاء از کار افتاده به n-k+1 عدد برسد.

اگر منابع برای تعمیر اجزاء خراب شده در اختیار قرار گیرد، می‌باید برای مدت بیشتری سیستم را سالم یا در حال کار نگه داشت. از این طریق ما انتظار داریم که چرخه[1] عمر سیستم را بیشتر کنیم. هر زمان تعداد اجزاء خراب در هر نقطه از زمان بیشتر از n-k باشد، چرخه عمر سیستم از کار می‌افتد(یحیی‌تبار، ۱۳۹۱)

[1] Cycle

۲-٤. تئوری صف – زنجیره‌های مارکف

۲-٤-۱ . فرآیند مارکف

برای روشن شدن مفهوم کلی فرآیند مارکف، می‌توان گفت که اگر زمان را در این فرآیند، به سه دوره گذشته، حال و آینده تقسیم کنیم، آینده این فرآیند بستگی به مسیری که در گذشته طی کرده است ندارد و تنها به موقعیت آن در زمان حال وابسته است.

فرآیند پواسون، برای مثال، نوعی فرآیند مارکف است، زیرا در آن تعداد پشامدهایی که از یک لحظه معین به بعد اتفاق می‌افتد، مستقل از پیشامدهایی است که قبل از آن افتاده است. به عبارت دیگر، چنانچه وضعیت فرآیند در لحظاتی مانند $t_1, t_2, \ldots, t_n$ مشخص باشد، می‌توان گفت که برای پیش بینی حرکت آینده این فرآیند، تنها آخرین اطلاعات، یعنی وضعیت فرآیند در لحظه t_n، کافی است.

خاصیت مارکفی یک فرآیند را می‌توان به زبان ریاضی زیر نیز بیان کرد. مجموعه‌ای ازمتغیرهای تصادفی $\{x(t), t \geq 0\}$ را درنظر بگیرید. اگر $x(t)$ طبق فرآیند مارکف عمل کند، به ازای تمام مقادیر $t_1 < t_2 < \ldots < t_n < t_{n+1}$ رابطه زیر برقرار است.

$$P[X(tn+1) \leq x | X(t1)=x1, X(t2)=x2, \ldots, X(tn)=xn] = P[X(tn+1) \leq x \mid X(tn)=xn] \quad (۲-٤٥)$$

فرآیندهای مارکف را به طورکلی برحسب دو عامل طبقه بندی می‌کنند:

الف: پارامتر t می‌تواند پیوسته یا گسسته باشد. گسسته بودن t را می‌توان چنین تفسیر کرد که رفتار سیستم تنها در مقاطع مشخصی از زمان مطالعه می‌شود. در صورتیکه t گسسته باشد، $x(t)$ با متغیر تصادفی $X_1, X_2, \ldots, X_n$ جایگزین می‌شود.

ب. مجموعه مقادیری را که $x(t)$ می‌تواند انتخاب کند، برحسب تعریف، حالت سیستم می‌نامند. حالت سیستم می‌تواند گسسته یا پیوسته باشد(مدرس و همکاران، ۱۳۹۱).

۲-٤-۲ . زنجیره‌های مارکف

زنجیره‌های مارکف حالت خاصی از فرآیند مارکف است که در آن هم پارامتر t و هم حالت سیستم فقط مقادیر گسسته را انتخاب می‌کنند. براین اساس یک رشته متغیر تصادفی $X_1 , X_2 , \dots , X_n$ را زنجیره مارکف می‌نامند، اگر به ازای تمام مقادیر n و تمام حالت‌های i و j رابطه زیر برقرار باشد.

$$P[X_{n+1}=j \mid X_1=i_1 , X_2=i_2 , \dots , X_n=i]=p[X_{n+1}=j \mid X_n=i] \qquad (٤٦-٢)$$

در اینجا به n، مرحله نیز گفته می‌شود.

برای مثال نقطه‌ای فیزیکی را در نظر بگیرید که روی خط مستقیم حرکت می‌کند. در هر حرکت، این نقطه به احتمال p به اندازه یک قدم به جلو و به احتمال (1-p) به همین اندازه به عقب می‌رود. اگر x(t)، یعنی حالت سیستم را موقعیت این نقطه روی خط در نظر بگیریم، x(t) یک زنجیره مارکف تشکیل می‌دهند زیرا موقعیت نقطه در حرکت‌های بعدی تنها به وضعیت فعلی آن بستگی دارد و مستقل از مسیری است که در گذشته طی کرده است (این فرآیند را قدم زدن تصادفی نیز می‌گویند)

احتمال تغییر حالت سیستم از i به j، یا به عبارت دیگر احتمال حرکت نقطه روی خط به شرح زیر است:

$$P[X_{n+1}=i-1 \mid X_n=i]=1-p, \qquad P[X_{n+1}=i+1 \mid X_n=i]= p \qquad (٤٧-٢)$$

$$P[X_{n+1}=j \mid X_n=i]=0 \qquad \text{و اگر } j\neq i+1 \text{ و یا } j\neq i-1 \text{ باشد،} \qquad (٤٨-٢)$$

۲-٤-۳ . ماتریس گذار

ماتریس گذار، ماتریسی است که p_{ij}، عنصر تشکیل دهنده آن در سطر i و ستون j (احتمال تغییر حالت سیستم از i به j) باشد. اگر فرض کنیم که تعداد حالت‌های سیستم M است،ماتریس گذار آن به شکل زیر در می‌آید.

$$p = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1M} \\ p_{21} & p_{22} & & p_{2M} \\ \vdots & & \ddots & \vdots \\ p_{M1} & p_{M2} & \cdots & p_{MM} \end{pmatrix} \qquad (٤٩-٢)$$

بدیهی است که تمام عناصر این ماتریس، غیر منفی و مجموع عناصر هر سطر برابر با یک است (مجموع عناصر یک ستون لزوما یک نیست). اگر تعداد حالت‌های سیستم نامتناهی باشد ابعاد ماتریس نیز نامتناهی خواهد بود. به

طور کلی، هر ماتریس مربع که تمام عناصر آن غیر منفی و مجموع عناصر هر سطر آن برابر با یک باشد را ماتریس مارکف می‌نامند(مدرس و همکاران، ۱۳۹۱).

برای مثال فرض کنید باریدن و نباریدن باران در روز آینده فقط بستگی به شرایط آب و هوایی امروز داشته باشد، اگر امروز بارانی باشد فردا با احتمال α=۰.۷ بارانی خواهد بود، و اگر امروز بارانی نباشد فردا با احتمال β=۰.٤ بارانی است که ماتریس گذار (انتقال حالت) آن بصورت زیر خواهد بود.

$$p = \begin{bmatrix} \alpha & 1-\alpha \\ \beta & 1-\beta \end{bmatrix} = \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}$$

۲-٤-٤ . گذار m مرحله‌ای

با استفاده از ماتریس گذار (انتقال حالت)، رابطه بین حالت‌های سیستم در دو مرحله متوالی مشخص می‌شود. به عبارت دیگر، با دانستن حالت هر مرحله می‌توان تابع توزیع حالت سیستم را در مرحله بعد تعیین کرد. در این قسمت می‌خواهیم تعیین کنیم که چه رابطه‌ای بین حالت سیستم در یک مرحله و حالت آن در m مرحله بعد موجود است. به عبارت دیگر، سئوالی که مطرح می‌شود این است که اگر در مرحله‌ای، حالت سیستم i باشد (یعنی X_n=i)، احتمال این که پس از m مرحله حالت سیستم به j برسد(X_{n+m}=j)، چقدر است؟

با داشتن ماتریس گذار، می‌توان تابع توزیع X_{n+1} را بدست آورد. با دانستن حالت مرحله n+1، تابع توزیع X_{n+2} و بهمین ترتیب X_{n+3}, ... ,X_{n+m} بدست می‌آید. لیکن، هدف این است که بتوان رابطه مستقیم بین X_n و X_{n+m} را مستقیما تعیین کرد. برای این منظور ابتدا تعاریف زیر را انجام می‌دهیم.

تعریف الف: $p_{ij}^{(m)}$ احتمال تغییر حالت سیستم از i به j در m مرحله است، یعنی:

$$p_{ij}^{(m)} = p[X_m = j \mid X_0 = i] \qquad (۵۰-۲)$$

ب: ماتریس گذار m مرحله‌ای ماتریسی است که اجزای آن $p_{ij}^{(m)}$ باشد. یعنی:

$$p^{(m)} = \begin{bmatrix} p_{11}^{(m)} & \cdots & p_{1M}^{(m)} \\ \vdots & \ddots & \vdots \\ p_{M1}^{(m)} & \cdots & p_{MM}^{(m)} \end{bmatrix} \qquad (۵۱-۲)$$

بنابراین برای تعیین رابطه بین حالت‌های سیستم در m مرحله، شناخت $p^{(m)}$ ضروری است. برای محاسبه $p^{(m)}$ رابطه‌های زیر را می‌توان بکار گرفت:

الف: به ازای تمام حالت‌های i و j و برای هر k و k' رابطه زیر برقرار است:

$$p_{ij}^{(k+k')} = \sum_{s=0}^{\infty} p_{is}^{k}\, p_{sj}^{k'} \qquad (۵۲-۲)$$

ب:

$$P^{(m)}=p.p \ldots . p=p^{m} \qquad (۵۳-۲)$$

رابطه (۲-۵۲) که به رابطه (C-K)[1] معروف است، بیان کننده این حقیقت است کـه احتمـال تغییـر حالـت‌هـای سیستم از i به j در $(k+k')$ مرحله ، برابر با مجموع تغییر حالت‌های سیستم از i به هر حالـت دیگـر سیسـتم،مثلا s در k مرحله و سپس از s به j در k' مرحله است. برای مثال زنجیره مارکفی را در نظر بگیرید که ماتریس گـذار آن به شرح زیر باشد (حالت‌های سیستم را ۱و ۲ و ۳ فرض کنید)

$$p = \begin{bmatrix} 0.75 & 0.25 & 0 \\ 0.25 & 0.5 & 0.25 \\ 0 & 0.75 & 0.25 \end{bmatrix}$$

مقادیر $p_{13}^{(2)}$(احتمال این که یک سیستم از حالت ۱ شروع و پس از ۲ مرحله به حالـت ۳ برسـد) و همچنـین $p_{31}^{(2)}$ بصورت زیر خواهد بود.

$$p^2 = p.p = \begin{bmatrix} 0.625 & 0.3125 & 0.0625 \\ 0.3125 & 0.5 & 0.1875 \\ 0.1875 & 0.5625 & 0.25 \end{bmatrix}$$

بنابراین:

$$p_{13}^{(2)} = 0.0625\,, p_{31}^{(2)} = 0.1875$$

مقادیر فوق را می‌توان مستقیما، با استفاده از رابطه‌های (۲-۵۲) نیز بدست آورد. مثلا

$$p_{13}^{(2)} = p_{11}p_{13} + p_{12}p_{23} + p_{13}p_{33} = 0.0625$$

که در واقع این حاصل ضرب سطر اول در ستون ماتریس گذار است.

۲-٤-٥ . احتمالات حدی در زنجیره مارکف

یکی از ویژگی‌های زنجیره مارکف این است که تحت شرایطی، تابع توزیـع حالـت سیسـتم در دراز مـدت بـه حالت اولیه آن وابسته نیست. برای روشن شدن موضوع، $p_{ik}^{(m)}$ و $p_{jk}^{(m)}$ را در نظر بگیرید. این دو مقدار در حالـت کلی با یک دیگر متفاوت هستند. به عبارت دیگر، احتمال این که سیسـتم پـس از m مرحلـه بـه حالـت k برسـد، بستگی به حالت فعلی آن دارد. اما تحت شرایطی، صرف نظر از این که سیستم از چه حالتی شـروع کنـد، احتمـال

[1] Chapman-kolomogrov

این که در دراز مدت به حالت k برسد مقداری ثابت است که این مقدار را با π_k نشان می‌دهند. به عبارت دیگر، صرف نظر از این که سیستم از هر حالتی مثلا i، شروع کند، خواهیم داشت:

$$\pi_k = \lim_{n\to\infty} p_{ik}^{(m)} \qquad (۵۴-۲)$$

به همین ترتیب، بدیهی است که اگر حرکت سیستم در دراز مدت مستقل از حالت شروع آن باشد، رابطه زیر نیز صدق می‌کند.

$$\pi_k = \lim_{m\to\infty} p(X_n = k) = \lim_{n\to\infty} \pi_k^n \qquad (۵۵-۲)$$

برای مثال زنجیره مارکف زیر را در نظر بگیرید:

$$p = \begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \end{bmatrix}$$

ماتریس‌های گذار ۲ و ٤ و ۸ مرحله‌ای این زنجیره مارکف عبارت است از:

$$p^2 = \begin{bmatrix} 0.52 & 0.48 \\ 0.36 & 0.64 \end{bmatrix} \text{ و } p^4 = \begin{bmatrix} 0.4433 & 0.5568 \\ 0.4176 & 0.5824 \end{bmatrix} \text{ و } p^8 = \begin{bmatrix} 0.4289 & 0.5711 \\ 0.4283 & 0.5717 \end{bmatrix}$$

همانطور که مشاهده می‌شود، صرف نظر از این که از چه حالتی شروع کنیم، به تدریج عناصر یک ستون به یک دیگر نزدیک می‌شوند و در نهایت با احتمال ثابتی به حالت‌های ۱یا ۲ می‌رسیم.در حالتی که احتمالات حدی وجود داشته باشد موقعی که $n\to\infty$ ماتریس p^n هم به ماتریسی میل می‌کند که تمام عناصر هر ستون آن، مثلا ستون iام برابر π_j خواهد شد.

با توجه به نکات گفته شده یک زنجیره مارکف با حالت نادوره‌ای را در نظر بگیرید. در این زنجیره مارکف:

الف. احتمالات حدی همیشه وجود دارد، یعنی صرفنظر از این که سیستم از چه حالتی شروع کند، در دراز مدت با احتمال ثابت π_j به حالت j گذار می‌کند.

ب. اگر حالت‌های سیستم برگشت پذیر مثبت باشد، به ازای تمام حالت‌ها، π_j عددی مثبت است.

ج. اگر حالت‌های سیستم گذار ویا برگشت پذیر تهی باشند، به ازای تمام حالت‌ها π_j، برابر با صفر است.

و همچنین در یک سیستم یکپارچه با حالت‌های برگشت پذیر مثبت و نادوره‌ای(که آنرا Ergodic می‌نامند) مقادیر احتمالات حدی π_j با حل دستگاه معادلات زیر بدست می‌آید.

$$\pi_j = \sum_{i=0}^{\infty} \pi_i p_{ij} \qquad j = 0,1,2,\dots \qquad (۵۶-۲)$$

یا

$$(\pi_0, \pi_1, \pi_2, \dots) = (\pi_0, \pi_1, \pi_2, \dots)p$$

$$\sum_{i=0}^{\infty} \pi_i = 1 \qquad (۵۷-۲)$$

که p ماتریس گذار زنجیره مارکف است. مقادیر π_j را، که از رابطه فوق بدست می‌آید، اصطلاحا توزیع ثابت زنجیره نیز می‌گویند.(باید توجه داشت که دستگاه معادلات فوق دارای یک معادله زاید است و لذا پس از حذف یکی از معادلات(۲-۵۶) تعداد معادلات و مجهول‌ها برابر خواهد شد).

زنجیره مارکف مثال قبل را درنظر بگیرید. این سیستم یکپارچه و نادوره‌ای است. ضمنا چون تعداد حالت‌های آن متناهی است، لذا تمام حالت‌ها برگشت پذیر مثبت هستند. احتمالات حدی را می‌توان طبق روابط (۲-۵۶) و (۲-۵۷) محاسبه کرد. یعنی:

$$(\pi_1, \pi_2) = (\pi_1, \pi_2) \begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \end{bmatrix}$$

و

$$\pi_1 + \pi_2 = 1$$

از حل معادلات فوق نتیجه می‌شود که: π_1=۰.٤۲۸٦ و π_2=۰.۵۷۱٤. π_1 و π_2 بترتیب معرف درصدی از اوقات است که سیستم در دراز مدت در حالت ۱ یا ۲ توقف می‌کند(مدرس و همکاران، ۱۳۹۱).

۲-٤-٦ . زنجیره‌های مارکف با زمان پیوسته

همانطور که در ابتدای این بخش گفته شد، در یک فرآیند مارکف هم پارامتر t و هم حالت سیستم X(t) می‌توانند، گسسته و یا پیوسته باشند. در یک زنجیره مارکف هر دوعامل فوق گسسته هستند(و به همین دلیل، به آنها زنجیره‌های مارکف با زمان گسسته نیز می‌گویند). زنجیره‌های مارکف با زمان پیوسته، حالت خاصی از فرآیند مارکف هستندکه در انها پارامتر t پیوسته است. اما فرض می‌شود که X(t)، حالت سیستم همچنان گسسته است. در زنجیره مارکف با زمان گسسته فرض می‌شود که تغییر حالت سیستم فقط در انتهای هر مرحله صورت می‌گیرد(و یا این که فقط در انتهای هر مرحله حالت سیستم مشاهده می‌شود)، در حالیکه در زنجیره‌های مارکف با زمان پیوسته چنین محدودیتی وجود ندارد. در نتیجه،حالت سیستم می‌تواند در هرلحظه تغییر کند.

تعریف. فرآیند مارکف $\{ X(t)\ t \geq ۰ \}$ رازنجیره مارکف با زمان پیوسته(یا به اختصار زنجیره‌های مارکف پیوسته)می‌نامند،اگر به ازای تمام مقادیر s و t رابطه زیر برقرار باشد:

$$p[X(t+s) = j \mid X(s) = i, X(u) = u(x), u \leq s] = p[X(t+s) = j \mid X(s) = i] \qquad (۵۸-۲)$$

اگر رابطه فوق مستقل از s باشد، این گونه زنجیره مارکف را همگن می‌نامند.به عبارت دیگر، احتمال گذار از یک حالت به حالت دیگر، به ازای تمام مقادیر i و j و t از رابطه زیر بدست می‌آید:

$$p_{ij} = p[X(t+s) = j \mid X(s) = i] = p[X(t) = j \mid X(0) = i] \qquad (۵۹-۲)$$

به این ترتیب، $p_{ij}(t)$ احتمال گذار سیستم از حالت i به j در مدت t است.

شکل ۲-۸ (گذار سیستم به حالت های مختلف نسبت به زمان در زنجیره مارکف پیوسته) حرکت سیستم به حالت‌های گوناگون را نسبت به زمان نشان می‌دهند. همانطور که مشاهده می‌شود، در هر لحظه از زمان، سیستم می‌تواند یکی از حالت‌های ممکن را انتخاب کند. در هر حالت، سیستم مدتی متوقف و سپس به حالت دیگر گذار می‌کند. مدت زمان توقف در هر حالت متغیری تصادفی است(مدرس و همکاران، ۱۳۹۱).

شکل ۲-۸ (گذار سیستم به حالت های مختلف نسبت به زمان در زنجیره مارکف پیوسته(مدرس و همکاران، ۱۳۹۱))

۲-٤-۷ . ماتریس گذار و ماتریس آهنگ گذار

در مورد زنجیره‌های مارکف پیوسته، دو نوع ماتریس را در مورد گذار حالت‌ها می‌توان معرفی کرد، یکی از آنها که ماتریس گذار نامیده و با p(t) نشان داده می‌شود دارای همان مفهوم ماتریس گذار در زنجیره‌های مارکف گسسته است. هر عنصر این ماتریس، $p_{ij}(t)$، معرف احتمال گذار سیستم از i به j در مدت t است. به این ترتیب، این ماتریس خود تابعی از پارامتر t خواهد بود.

حالت خاص این ماتریس، p(0)، با توجه به عناصر تشکیل دهنده آن به شرح زیر است:

$$P_{ii}(0)=1$$

و به ازای $j \neq i$ ،

$$P_{ij}(0)=0$$

بنابراین، این ماتریس بشکل ماتریس یکه در می‌آید،یعنی، $p(0)=I$

ماتریس دیگری که می‌تواند گذار حالت‌ها را بیان کند، ماتریس آهنگ گذار است،که با Q نشـان داده مـی‌شـود. عناصر تشکیل دهنده این ماتریس، q_{ij}، به شرح زیر تعریف می‌شود:

الف: اگر $i \neq j$ باشد،

$$q_{ij} = lim_{t\to\infty}\frac{p_{ij(t)}}{t} = \frac{d}{dt}p_{ij}(t) \qquad (۶۰\text{-}۲)$$

با در نظر گرفتن این که $p_{ij}(t)$ معرف احتمال تغییر حالت سیستم از i به j در مدت زمان t است. q_{ij} نشان دهنده آهنگ تغییر سیستم از i به j خواهد بود.

ب. اگر $i=j$ باشد،

$$q_{ii} = -\sum_{i\neq j} q_{ij} \qquad (۶۱\text{-}۲)$$

از طرف دیگر، رابطه (۲-۶۱) را با کمک رابطه (۲-۶۰) می‌توان به شرح زیر بیان کرد:

$$q_{ii} = -\sum_{i\neq j}\frac{d}{dt}p_{ij}(\mathrm{t}) = \frac{d}{dt}[-1 + p_{ii}(t)] = \frac{d}{dt}p_{ii}(\mathrm{t}) \qquad (۶۲\text{-}۲)$$

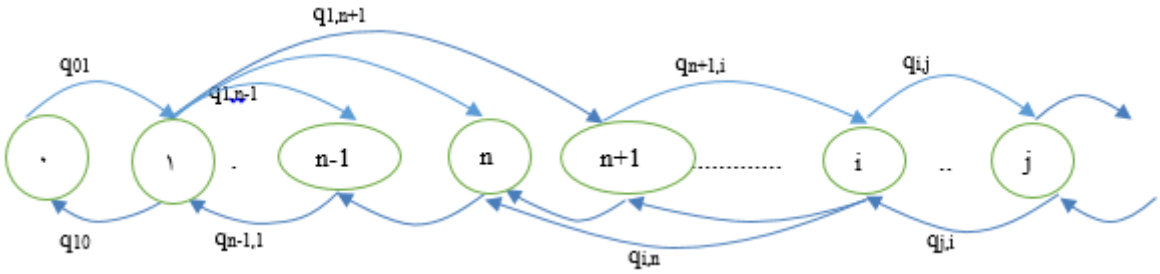
به این ترتیب، عناصر قطری ماتریس Q همگی منفی هستند و مجموع عناصر هر سطر برابر با صفر خواهد بود. با توجه به رابطه (۲-۶۱)، ($-q_{ii}$) معرف آهنگ تغییر حالت از i به مجموعه حالت‌های دیگر و q_{ii} معـرف آهنـگ ماندن سیستم در i است.

ارتباط ماتریس آهنگ گذار با ماتریس گذار را می‌توان با استفاده از رابطه زیر مشخص ساخت:

$$Q = \lim_{t\to\infty}\frac{p(t)-I}{t} = \frac{d}{dt}p(t)\mid_{t=0} \qquad (۶۳\text{-}۲)$$

۲-۴-۸ . نمودار آهنگ

رابطه بین حالت‌های یک سیستم مارکفی را می‌توان با نمودار آهنگ نیز نشان داد شکل ۲-۹(نمودار آهنگ یـک سیستم مارکوف). در این نمودار، گره‌ها معرف حالت سیستم و شاخه‌ها نشان دهنده امکان گذار از هـر حالـت بـه حالت‌های دیگر است. مقادیر روی شاخه‌ها، q_{ij}، بر اساس تعریف فوق، نشان دهنده آهنگ تغییر حالت سیستم از i به j است. در نمودار آهنگ، q_{ii}ها نشان داده نمی‌شوند زیرا می‌توان مقدار آنرا با توجـه بـه رابطـه (۲-۶۱) بدسـت آورد(مدرس و همکاران، ۱۳۹۱).



شکل ۲-۹(نمودار آهنگ یک سیستم مارکوف)

۲-٤-۹ . روابط حدی در زنجیره مارکف با زمان پیوسته

تابع توزیع حالت سیستم در دراز مدت،($t\to\infty$)، را می‌توان با استفاده از قضیه زیر که مشابه آن در زنجیره مارکف گسسته اشاره شد بدست آورد.

قضیه: اگر سیستم یکپارچه باشد، در این صورت حد $p_{ij}(t)$ به سمت عددی میل می‌کند که آنرا π_j می‌نامیم، یعنی $\lim_{t\to\infty} p_{ij}(t) = \pi_j$

ضمنا اگر تمام حالات برگشت پذیر مثبت باشند، که در این صورت سیستم را ارگودیک[1] می‌نامند، مقدار احتمال حدی نیز از رابطه زیر بدست می‌آید:

$$\sum_j \pi_j q_{ji} = 0, \quad i = 1,2,\dots \;\&\; \sum_j \pi_j = 1 \qquad (٦٤-٢)$$

رابطه (٢-٦٤)را به شکل ماتریسی نیز می‌توان نشان داد:

$$[\pi_1, \pi_2, \dots]Q = 0$$

که Q ماتریس آهنگ گذار و π_j معرف درصدی از زمان است که سیستم در حالت j می‌ماند.

رابطه (٢-٦٤)را می‌توان با کمک نمودار آهنگ نیز بدست آورد. برای هر حالت، مثلا i، رابطه فوق را به شکل زیر در می‌آوریم:

$$\pi_i q_{ii} = -\sum_{j\neq i} \pi_i q_{ji} \;, \qquad i = 1,2,\dots.$$

اما می‌دانیم که، $q_{ii} = -\sum_{j\neq i} q_{ji}$

درنتیجه درگره i،

$$\sum_{j\neq i} \pi_i q_{ij} = \sum_{j\neq i} \pi_j q_{ji}$$

[1] Ergodic

جملات سمت چپ مربوط به آهنگ گذار خروجی از i به بقیه حالت‌ها و جملات سمت راست مربوط به آهنگ گذار از سایر حالت‌ها به i است. هر جمله حاصل ضرب دو کمیت است. اولی آهنگ گذار از یک حالت به دیگر(q_{ij})، و دیگری احتمال بودن سیستم در حالت مبدا(π_j)، است. بدیهی است که، با توجه به این که رابطه (۲-٦٤) برای بدست آوردن رابطه حدی است، مقادیر احتمالات مورد نظر نیز مربوط به دراز مدت خواهد بود.

به این ترتیب، برای محاسبه روابط حدی می‌توان از نمودار آهنگ استفاده کرد. برای این منظور مجموع $\pi_n q_{ji}$های خروجی و ورودی هر حالت را مساوی یک دیگر قرار می‌دهیم. اینگونه معادلات را معادلات تعادلی نیز می‌گویند(مدرس و همکاران، ۱۳۹۱).

برای مثال زنجیره مارکفی که ماتریس اهنگ گذار آ ن به شرح زیر است را درنظر بگیرید.

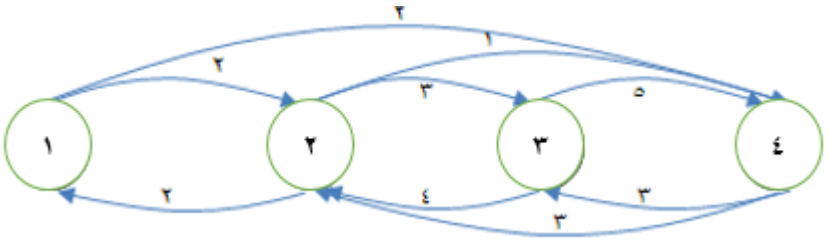
$$Q = \begin{bmatrix} -4 & 2 & 0 & 2 \\ 2 & -6 & 3 & 1 \\ 0 & 4 & -9 & 5 \\ 0 & 3 & 3 & -6 \end{bmatrix}$$

رابطه‌های حدی برای این زنجیره به شرح زیر محاسبه می‌شود.

$$[\pi_1, \pi_2, \pi_3, \pi_4]Q = 0$$

$$\begin{cases} -4\pi_1 + 2\pi_2 = 0 \\ 2\pi_1 - 6\pi_2 + 4\pi_3 + 3\pi_4 = 0 \\ 3\pi_2 - 9\pi_3 + 3\pi_4 = 0 \\ 2\pi_1 + \pi_2 + 5\pi_3 - 6\pi_4 = 0 \\ \pi_1 + \pi_2 + \pi_3 + \pi_4 = 0 \end{cases}$$

معادلات فوق را می‌توان از نمودار آهنگ نیز بدست آوریم. نمودار آهنگ این مثال در شکل ۲-۱۰(نمودار آهنگ یک سیستم مارکوف) تشان داده شده است.



شکل ۲-۱۰(نمودار آهنگ یک سیستم مارکوف)

معادلات تعادلی را برای هر گره(حالت)، می‌نویسیم:

گره ۱:

مقادیر مربوط به خروج از گره ۱، π_1(۲+۲) و مقادیر مربوط به ورود گره ۱، $2\pi_2$

در تنیجه با استفاده از معادله تعادلی داریم: $4\pi_1 = 2\pi_2$

گره ۲: $(2+3+1)\pi_2 = 2\pi_1 + 4\pi_3 + 3\pi_4$

گره ۳: $(4+5)\pi_3 = 3\pi_2 + 3\pi_4$

گره ٤ $(3+3)\pi_4 = 2\pi_1 + \pi_2 + 5\pi_3$

همانطور که مشاهده می‌شود، معادلات فوق همان معادلات قبلی هستند، که مستقیما از ماتریس آهنگ گذار بدست آمدند.

۲-٤-۱۰ . فرآیند تولد و مرگ

برای تشریح این فرآیند، جمعیتی را در نظر بگیرید که تعداد آن هر لحظه می‌تواند افزایش یابد(تولد) و یا ازآن کاسته گردد(مرگ). در یک سیستم صف، مشتری‌های داخل سیستم معرف جمعیت فوق الذکر است. در چارچوب فرآیند تولد و مرگ، ورود مشتری جدید در حکم تولد و خروج مشتری جدید (هر مشتری پس از دریافت خدمت)، در حکم مرگ محسوب می‌شود. ضمنا، ویژگی عمده فعالیت تولد و مرگ این است که فاصله زمانی بین دو اتفاق (تولد یا مرگ) بر اساس توزیع نمایی است. به عبارت دیگر، در این فرآیند، تعداد تولدها و یا مرگ‌ها در یک فاصله زمانی مشخص بر اساس فرآیند پواسون است. به بیان دقیق‌تر، فرآیند تولد و مرگ براساس فرض‌های زیر انجام می‌شود.

فرض تولد. یک لحظه مشخص را درنظر بگیرید که جمعیت سیستم در آن برابر n باشد. مدت زمانی که طول می‌کشد تا یک تولد صورت گیرد(یک مشتری جدید وارد شود)، متغیر تصادفی با توزیع نمایی و پارامتر λ_nاست. به عبارت دیگر، احتمال ورود مشتری در فاصله کوتاه Δt هم متناسب با طول Δt و هم متناسب با λ_n است. به بیان ریاضی مقدار احتمال برابر است با $\lambda_n\ \Delta t+0(\Delta t)$ به همین ترتیب احتمال ورود بیش از یک مشتری در فاصله کوتاه Δt برابر با $0(\Delta t)$ است. به عبارت دیگر، با این فرض دو تولد همزمان امکان ندارد. آهنگ ورود مشتری (یا آهنگ تولد)، یعنی λ_n، می‌تواند به جمعیت سیستم بستگی داشته باشد، ولی مستقل از زمان فرض شود. $\lambda 0$ لزوما برابر با صفر نیست.

فرض مرگ. یک لحظه مشخص را در نظر بگیرید که در آن جمعیت سیستم برابر n باشد. مدت زمانی که طول می‌کشد تا یک مرگ صورت گیرد (یا یک مشتری خارج شود)، متغیری تصادفی با توزیع نمایی و پارامتر μ_n است. آهنگ خروج مشتری (یا مرگ)، یعنی μ_n، نیز می‌تواند به جمعیت سیستم بستگی داشته باشد، ولی مستقل از زمان فرض می‌شود. μ_0 برابر با صفر است زیرا جمعیت منفی معنا ندارد.

بنابراین نتیجه می‌شود که در فرآیند تولد و مرگ، در هرلحظه حداکثر یک مشتری وارد یا خارج می‌شود. علت این امر ناشی از خاصیت فرآیند پواسون است.

آهنگ تولد (یا ورود مشتری) یعنی λ_n، و همچنین آهنگ مرگ یعنی μ_n، در سیستم‌های مختلف می‌تواند متفاوت باشد. یک حالت خاص، $\lambda_n = \lambda$ یا $\mu_n = \mu$ (به ازای n=1,2,...) است. در حالت خاص دیگر، $\lambda_n = n\,\lambda$ ویا $\mu_n = n\,\mu$ است. در این حالت، متناسب با افزایش جمعیت، آهنگ ورود و یا خروج نیز افزایش می‌یابد.

طبق فرض‌های فوق، در مدت زمان کوتاه، جمعیت مشتری‌های داخل سیستم حداکثر یک نفر افزایش یا کاهش می‌یابد، مشروط بر این که طبق فرآیند تولد و مرگ عمل کند(مدرس و همکاران، ۱۳۹۱).

۲-٤-۱۱ .بیان فرآیند تولد و مرگ در چارچوب زنجیره مارکوف

در هر لحظه، حالت سیستم را تعداد مشتری داخل سیستم معرفی می‌کنیم. طیق خاصیت فرآیند تولد و مرگ، افزایش و کاهش این تعداد مشتری، فقط بستگی به حالت سیستم دارد و مستقل از گذشته آن است. بنابراین، خاصیت مارکفی بودن در مورد این فرآیند صادق است. ضمنا مدت زمانی هم که طول می‌کشد تا حالت سیستم تغییر کند، متغیری تصادفی با توزیع نمایی است. به این ترتیب، این فرآیند در چارچوب زنجیره مارکف پیوسته قرار می‌گیرد.

برای مشخص کردن ماتریس آهنگ گذار، باید عناصر این ماتریس، یعنی q_{ij} را به ازای تمام مقادیر i و j محاسبه کرد. می‌دانیم که طبق تعریف، به ازای i≠j،

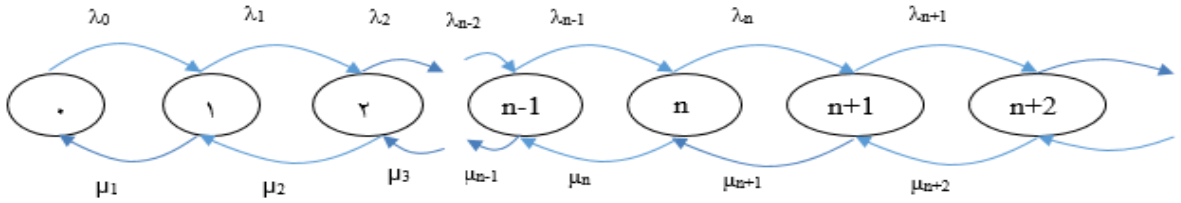
$$q_{ij} = \lim_{\Delta t \to 0} \frac{p\left(\text{تغییر حالت سیستم از } i \text{ به } j \text{، در مدت } \Delta t\right)}{\Delta t}$$

بنابراین، $q_{i,i+1} = \lambda_i$ (تولد) و $q_{i,i-1} = \mu_i$ (مرگ) و در سایر موارد (به ازای j≠i+1 یا j≠i-1) $q_{ij}=0$ است. به این ترتیب، ماتریس آهنگ گذار فرآیند تولد و مرگ به شکل زیر در می‌آید.

$$Q = \begin{bmatrix} -\lambda_0 & \lambda_0 & 0 & 0 & 0 & \dots \\ \mu_1 & -(\mu_1+\lambda_1) & \lambda_1 & 0 & 0 & \dots \\ 0 & \mu_2 & -(\mu_2+\lambda_2) & \lambda_2 & 0 & \dots \\ \vdots & & & & & \end{bmatrix}$$

به جای استفاده از ماتریس فوق، می‌توان از نمودار آهنگ برای نشان دادن فرآیند مارکف استفاده کرد. چنین نمودار آهنگی در شکل ۲-۱۱(نمودار آهنگ در فرآیند تولد و مرگ) نشان داده شده است. همانطور که مشاهده

می‌شود، در فرآیند تولد و مرگ، از یک حالت، تنها می‌توان به حالت‌های مجاور آن حرکت کرد(مدرس و همکاران، ۱۳۹۱).



شکل ۲-۱۱(نمودار آهنگ در فرآیند تولد و مرگ(مدرس و همکاران، ۱۳۹۱))

۲-٤-۱۲ . معادلات تعادلی در سیستم‌های نمایی

همانطور که قبلا گفته شد، π_n معرف بودن n *مشتری در سیستم* یا درصدی از زمان است که دارای n مشتری است(در دراز مدت)، یا $\pi_n = \lim_{n\to\infty} p[N(t) = n]$

برای محاسبه π_n در فرآیند تولد و مرگ، شکل ۲-۱۱(نمودار آهنگ در فرآیند تولد و مرگ) بکار گرفته می‌شود. ابتدا از حالت صفر، که هیچ مشتری در سیستم وجود ندارد شروع می‌کنیم. همان‌طور که نمودار آهنگ نشان می‌دهد حالت سیستم فقط در صورتی تغییر می‌کند که یک مشتری جدید وارد شود. در این‌صورت، حالت سیستم نیز"۱" خواهد شد. تغییر حالت سیستم از صفر به یک (یعنی ورود مشتری) با آهنگ λ_0 انجام می‌شود. به همین ترتیب، با در نظر گرفتن این مطلب که ورود به حالت صفر نیز فقط با تغییر حالت سیستم از یک به صفر امکان پذیر است، آهنگ ورود به این حالت نیز برابر با μ_1 خواهد بود. در نتیجه، معادله تعادلی در حالت صفر برابر است با:

$$\lambda_0\pi_0 = \mu_1\pi_1 \qquad (۲-۶۵)$$

اکنون حالت ۱ را در نظر بگیرید؛ تغییر حالت سیستم به دو طریق ورود مشتری جدید (با آهنگ λ_1) یا خروج تنها مشتری داخل سیستم (با آهنگ μ_1) میسر است. از طرف دیگر، از دو طریق می‌توان به حالت ۱ رسید. از حالت صفر با ورود یک مشتری یا از حالت ۲ با خروج یکی از مشتری‌های داخل سیستم. بنابراین، معادله تعادلی برای این حالت عبارت است از:

$$(\lambda_1 + \mu_1)\pi_1 = \lambda_0\pi_0 + \mu_2\pi_2 \qquad (۲-۶۶)$$

به همین ترتیب، معادله تعادلی برای حالت $n \geq 1$ *به شرح زیر خواهد بود:*

$$(\lambda_n + \mu_n)\pi_n = \lambda_{n-1}\pi_{n-1} + \mu_{n+1}\pi_{n+1} \qquad (۲-۶۷)$$

با استفاده از رابطه‌های (۲-٦٥)، (۲-٦٦) و (۲-٦٧)نتیجه می‌شود:

$$\pi_1 = \frac{\lambda_0}{\mu_1}\pi_0$$
$$\pi_2 = \frac{\lambda_0\lambda_1}{\mu_1\mu_2}\pi_0$$

و به طور کلی خواهیم داشت:

$$\pi_n = \frac{\lambda_0\lambda_1\cdots\lambda_{n-1}}{\mu_1\mu_2\cdots\mu_n}\pi_0, \qquad n \geq 1$$

برای سهولت از قرارداد زیر استفاده می‌کنیم:

$$c_n = \frac{\lambda_0\lambda_1\cdots\lambda_{n-1}}{\mu_1\mu_2\cdots\mu_n}, \qquad n \geq 1 \qquad (۲-٦٨)$$

در نتیجه با استفاده از رابطه (۲-٦٨) خواهیم داشت:

$$\pi_n = c_n\pi_0\,, \qquad n \geq 1 \qquad (۲-٦٩)$$

از طرف دیگر می‌دانیم که $\sum_{n=0}^{\infty}\pi_n = 1$ یا $\pi_0 + \sum_{n=1}^{\infty} c_n\pi_0 = \pi_0[1 + \sum_{n=1}^{\infty} c_n] = 1$.

در نتیجه:

$$\pi_0 = \frac{1}{1+\sum_{n=1}^{\infty} c_n} \qquad (۲-۷۰)$$

به این ترتیب، با استفاده از رابطه‌های (۲-٦٨)، (۲-٦٩) و (۲-۷۰) تابع توزیع تعداد مشتری در سیستم، در دراز مدت، بدست می‌آید(مدرس و همکاران، ۱۳۹۱).

۲-۵. الگوریتم رقابت استعماری

۲-۵-۱ . مقدمه

الگوریتم های بهینه‌سازی الهام گرفته از طبیعت به عنوان روش‌های هوشمند بهینه‌سازی در کنار روش‌های کلاسیک، به‌طور چشمگیری موفقیت‌آمیز بوده‌اند. از جمله این روش‌ها می‌توان به الگوریتم‌های ژنتیک(GA)[1] (الهام گرفته از تکامل بیولوژیکی انسان و سایر موجودات)، بهینه‌سازی کلونی مورچه‌ها(ACO)[2] (بر مبنای حرکت بهینه مورچه‌ها) و روش تبرید شبیه‌سازی شده(SA)[3] (با الهام‌گیری از فرایند تبرید فلزات) اشاره کرد. این روش‌ها

[1] Genetic Algorithms (GA)
[2] Ant Colony Optimization (ACO)
[3] Simulated Annealing (SA)

در حل بسیاری از مسائل بهینه‌سازی در حوزه‌های مختلفی چون تعیین مسیر بهینه عامل‌های خودکار، طراحی بهینه کنترل کننده برای پروسه های صنعتی، حل مسائل پایه مهندسی و غیره کاربرد دارند.

الگوریتم‌های بهینه‌سازی معرفی شده، به طور عمده الهام گرفته از فرایندهای طبیعی هستند و در ارائه این الگوریتم‌ها به سایر نمودهای تکامل انسانی توجهی نشده است. در این قسمت الگوریتم جدیدی برای بهینه‌سازی مطرح می‌شود که نه از یک پدیده طبیعی، بلکه از یک پدیده اجتماعی – انسانی الهام گرفته است. این الگوریتم به‌طور ویژه به فرایند استعمار، به عنوان مرحله‌ای از تکامل اجتماعی – سیاسی بشر نگریسته و با مدل‌سازی ریاضی این پدیده تاریخی، از آن به عنوان منشأ الهام یک الگوریتم قدرتمند در زمینه بهینه‌سازی استفاده می‌کند. در مدت کوتاهی که از معرفی این الگوریتم می‌گذرد، از آن برای حل مسائل بسیاری در حوزه بهینه‌سازی استفاده شده است. طراحی چیدمان بهینه برای واحدهای صنعتی، آنتن‌های مخابراتی هوشمند، سیستم‌های پیشنهاددهنده هوشمند و نیز طراحی کنترل کننده بهینه برای سیستم‌های صنعتی شیمیایی، از معدود کاربردهای گسترده این الگوریتم در حل مسائل بهینه‌سازی است. به‌طور خلاصه الگوریتم معرفی شده، با چند کشور اوله شروع می‌شود. این کشورها به دسته‌هایی به نام امپراتوری تقسیم می‌شوند. هر امپراتوری از تعدادی مستعمره و یک امپریالیست تشکیل شده است. در داخل امپراتوری، سیاست جذب از سوی استعمارگران به مستعمرات اعمال شده، آنها را در زمینه محورهای مختلف اجتماعی – سیاسی به سوی خود می‌کشد. به همراه سیاست جذب، رقابتی نیز میان امپراتوری‌ها برقرار است و همه آنها برای در دست گرفتن مستعمرات یک دیگر تلاش می‌کنند. حاصل این چرخه جذب و رقابت، هم‌گرایی کشورها (جواب‌های ممکن مساله) به سمت نقطه بهینه مطلق است.

۲-۵-۲ . تعریف رقابت استماری

با در نظر گرفتن الگوریتم‌های بهینه‌سازی مطرح شده، این نکته جلب توجه می‌کند که بیشتر روش‌های بهینه‌سازی عمومی، شبیه‌سازی کامپیوتری فرایندهای **طبیعی** هستند. شاید یکی از دلایل این امر، ملموس بودن و سادگی محاسبات و درک تکامل این فرآیندها است. در نقطه مقابل، در ارائه‌ی الگویتم‌های بهینه‌سازی، علی‌رغم توجه به تکامل زیستی انسان و سایر موجودات (الگوریتم‌های ژنتیک و ...) به تکامل اجتماعی وتاریخی او به عنوان پیچیده‌ترین و موفق‌ترین حالت تکامل، توجه چندانی نشده است. در این بخش، یک الگوریتم الهام گرفته از تکامل اجتماعی انسان، به نام الگوریتم رقابت استعماری (ICA)[1] برای بهینه‌سازی، توسعه داده شده است. الگوریتم

[1] Imperialist Competitive Algorithm

جدید معرفی شده با الهام‌گیری از یک فرایند اجتماعی – سیاسی، نسبت به روش‌های مطرح شده توانایی بیشتر و سرعت بالاتری دارند.

الگوریتم رقابت استعماری، در حالت اولیه از چند کشور شروع می‌شود. کشورها در حقیقت جواب‌های ممکن مساله هستند و معادل <<کروموزوم>> در الگوریتم ژنتیک و <<ذره>> در بهینه سازی ازدحام ذرات، به حساب می‌آیند. همه کشورها به دو دسته تقسیم می‌شوند: استعمارگر(امپریالیست)[1] و مستعمره[2] کشورهای استعمارگر با اعمال سیاست جذب(همگون سازی) در محورهای مختلف بهینه سازی، کشورهای مستعمره را به سمت خود می-کشند. رقابت امپریالیستی در کنار سیاست همگون سازی، هسته اصلی این الگوریتم را تشکیل می‌دهند و باعث می‌شود کشورها به سمت کمینه مطلق تابع حرکت کند.

در این قسمت به استعمار به‌عنوان جزئی لاینفک ازسیر تکامل تاریخی انسان نگریسته شده و از چگونگی اثر گذاری آن بر کشورهای استمارگر و مستعمره ونیز کل تاریخ، به عنوان منبع الهام یک الگوریتم کارا و نو در زمینه محاسبات تکاملی یاد شده است (توکلی مقدم و همکاران،۱۳۹۲).

۲-۵-۳ . مروری تاریخی بر پدیده استعمار

امپریالیزم، در لغت به سیاست توسعه قدرت و نفوذ یک کشور در حوزه خارج از قلمرو شناخته شده برای آن، اطلاق می‌شود. یک کشور می‌تواند کشور دیگر را از راه قانون‌گذاری مستقیم، یا از طریق روش‌های غیر مستقیم، مثل نظارت بر کالاها و مواد خام آن، کنترل کند. مورد اخیر اغلب استعمار نو[3] نامیده می‌شود. استعمار یک پدیده ذاتی در تاریخ بوده است که در مراحل ابتدایی همان نفوذ سیاسی ـ نظامی در کشورها و به شکل استفاده صرف از منابع زمینی، انسانی و سیاسی بوده است. گاهی نیز استعمار، به صرف جلوگیری از نفوذ کشور استعمارگر رقیب انجام می‌شد. به هر حال کشورهای استعمارگر رقابت جدی برای به استعمار کشیدن مستعمرات یک دیگر داشتند. این رقابت به نوبه خود باعث رشد و توسعه کشورهای استعمارگر از لحاظ سیاسی، نظامی و اقتصادی می‌شد؛ زیرا کشورها برای داشتن امکان رقابت، ناگزیر از گسترش بودند.

---

[1] Imperialist
[2] Colony

[3] Neocolonialism

در حالت‌های ابتدایی‌تر، استعمارگران با بهره‌گیری از منابع زمینی، انسانی و غیره کشور مستعمره، فقط در پی افزایش قدرت خود بودند و پیشرفت یا رکود مستعمرات مهم نبود. اما بعدها با افزایش ارتباط میان ملل و رشد انسانی، استعمارگران برای ادامه نفوذ خود، به نوعی از پذیرش همگانی (حمایت مردمی) نیز احتیاج یافتند. به‌این ترتیب کشورهای استعمارگر- هرچند در ظاهر- شروع به عمران و آبادسازی مستعمراتشان کردند.درنتیجه مستعمرات در زمینه‌های اقتصادی، اجتماعی و انسانی رو به پیشرفت نهادند که عامل این پیشرفت اجباری، کشور استعمارگر بود.

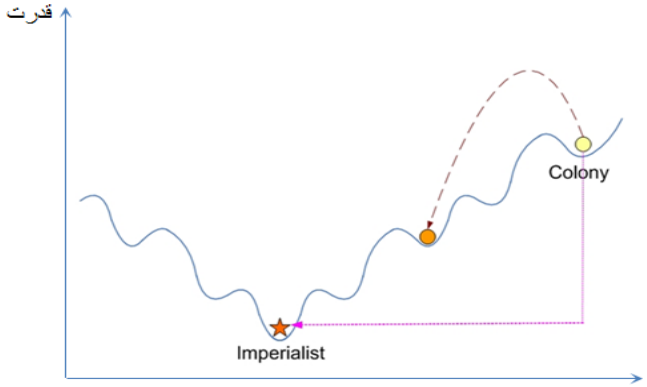
دلیل نام‌گذاری این فرایند با نام «استعمار» که ریشه در «عمران» و آبادی دارد، نیز همین مساله است. جلب اقبال عمومی تنها دلیلی نبود که به عمران مستعمرات توسط استعمارگران منجر می‌شد. دلیل دیگر، تسلط فرهنگی بر مستعمرات بود که این امر با اجرای سیاست همگون‌سازی امکان‌پذیرمی‌گشت. برای مثال، کشورهایی نظیر فرانسه و انگلیس به ایجاد مدارس انگلیسی و فرانسوی زبان در مستعمرات خود پرداختند. این اقدام به دلایل مختلفی صورت می‌گرفت که مهم‌ترین‌شان افزایش نفوذ فرهنگی در مستعمرات بوده است. ناگفته نماند که فرایند استعمار (حداقل بعد فرهنگی آن) با همه پیامدهای منفی، در بعضی از کشورهای امپریالیست حکم یک جهاد فکری برای نجات بشر را داشت. اشعاری دردست داریم با موضوع مدح و ستایش جوانان انگلیسی که با هدف آموزش راهی کشورهای مستعمره شده‌اند و از آنان به عنوان قهرمانان ملی در جبهه نجات بشری یاد می‌شود.

امپریالیزم، نگرش عمومی نسبت به تمدن غرب را تغییر داد. داروینیست‌های اجتماعی، امپریالیزم را تفسیر نمودند و ایده برتری فرهنگ غرب نسبت به فرهنگ شرق را تقویت کردند. در افریقا فقط آن افرادی که بعضی از استانداردهای فرهنگی غرب را داشتند، بخشی از حقوق اجتماعی خود را حفظ کردند. پرتغالی‌ها این مردم را جذب شده[1] و فرانسوی‌ها به‌طور توهین‌آمیزی آن‌ها را تکامل‌یافته[2] می‌نامیدند.

به هرحال جدا از آثار و تبعات مثبت و منفی آن، استعمار به عنوان یک فرایند ذاتی در تاریخ بشر ایجاد شد و در عین وارد کردن خسارت‌های جبران ناپذیر در بعضی موارد تاثیر مثبتی در نیز برای کشورها مستعمره (به‌ویژه زیربناهای فرهنگی)داشت. از دید بهینه‌سازی، استعمار بعضی از کشورها را که در یک دره معمولی تمدن قرار داشتند، خارج کرده و آنها را به یک حوزه کمینه دیگر برد که در بعضی موارد وضعیت این حوزه کمینه بهتر از موقعیت قبلی کشور مستعمره بود. با این حال، این حرکت مستلزم پیشروی مستعمره در جهت محورهای مختلف

[1] Assimilados
[2] Evolues

اقتصادی و فرهنگی به سمت یک امپریالیست قویتر بود؛ که این به‌معنی از میان رفتن بعضی از ساختارهای فرهنگی و اجتماعی است. شکل ۲-۱۲(اعمال سیاست جذب از سوی استعمارگران بر مستعمرات) این وضعیت را به‌خوبی نشان می‌دهد. در این شکل، مستعمره در نتیجه سیاست همگون‌سازی، از یک ناحیه کمینه خارج، و به یک ناحیه کمینه دیگر می‌شود که در آن وضعیت بهتری دارد. به هر حال هزینه‌ای که بابت این حرکت پرداخت شده است، نزدیکی به کشور استعمارگر در جهت محورهای مختلف اقتصادی، سیاسی و اجتماعی است. ادامه این حرکت می‌تواند به جذب کامل کشور مستعمره در کشور استعمارگر بیانجامد(توکلی مقدم و همکاران،۱۳۹۲).



شکل ۲-۱۲(اعمال سیاست جذب از سوی استعمارگران بر مستعمرات(آتش پزگرگری، ۱۳۸۷))

در بند بعد، این رابطه پیچیده اجتماعی در قالب یک الگوریتم بهینه‌سازی ریاضی ارائه می‌شود.

۲-۵-٤ . الگوریتم رقابت استعماری

الگوریتم رقابت استعماری، همانند دیگر الگوریتم‌های تکاملی، با تعدادی جمعیت اولیه تصادفی، که هر کدام از آنها یک کشور نامیده می‌شوند؛ شروع می‌شود. تعدادی از بهترین عناصر جمعیت (معادل نخبه‌ها در الگوریتم ژنتیک) به عنوان امپریالیست انتخاب می‌شوند. باقیمانده جمعیت نیز درحکم مستعمره هستند. استعمارگران بسته به قدرتشان، این مستعمرات را با یک روند خاص که در ادامه می‌آید؛ به سمت خود می‌کشند. قدرت کل هر امپراتوری، به هر دو بخش تشکیل‌دهنده آن، یعنی کشور امپریالیست (به عنوان هسته مرکزی) و مستعمرات آن، بستگی دارد.

در حالت ریاضی، این وابستگی با تعریف قدرت امپراتوری به صورت مجموع قدرت کشور امپریالیست، به اضافه در صدی از میانگین قدرت مستعمرات آن، مدل شده است. در شکل ۲-۱۳(شبه کد الگوریتم عمومی رقابت استعماری) شبه کد الگوریتم رقابت استعماری نشان داده شده است.

تعدادی از کشورها (اعضای جمعیت) را بطور تصادفی انتخاب کنید.
بهترین کشورها (اعضای جمعیت) را به‌عنوان کشورهای امپریالیست(استعمارگر) در نظر بگیرید.
کشورهای مستعمره (اعضای جمعیت) را به کشورهای امپریالیست تخصیص دهید(سیاست همسان سازی).
تا هنگامی که فقط یک امپراتوری باقی مانده باشد مراحل زیر را **انجام دهید:**
برای هر امپراتوری شامل کشور امپریالیست ومستعمره‌ها مراحل زیر را **انجام دهید:**
اگر مستعمره‌ای در امپراتوری وجود داشته باشد که هزینه آن از امپریالیستش کمتر باشد **آنگاه:**
جای آن کلونی را با کشور امپریالیستش عوض کرده و سپس هزینه کل آن امپراتوری را با امپریالیست جدیدش محاسبه کنید.
در غیر اینصورت:
هزینه کل امپراتوری را درحالت فعلی محاسبه کنید.
پایان.
ضعیف‌ترین مستعمره از ضعیف‌ترین امپراتور را انتخاب کرده و آن را به نزدیک‌ترین امپراتوری که بیشترین احتمال تصاحب را دارد، انتقال دهید.
اگر مستعمره دیگری در امپراتوری وجود نداشت،**آنگاه**:
آن امپراتوری را حذف کنید.
پایان تا هنگامی که.

شکل ۲-۱۳(شبه کد الگوریتم عمومی رقابت استعماری)

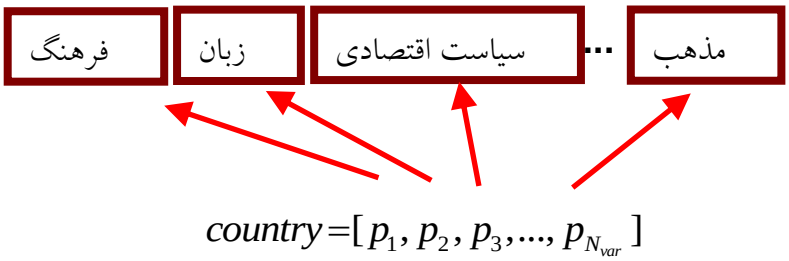
با شکل‌گیری امپراتوری‌های اولیه، رقابت امپریالیستی میان آن‌ها شروع می‌شود. هر امپراتوری‌ای که نتواند در رقابت استعماری، موفق عمل کرده و بر قدرت خود بیفزاید (یا حداقل از کاهش نفوذش جلوگیری کند)، از صحنه رقابت استعماری، حذف خواهد شد. بنابراین بقای یک امپراتوری به قدرت آن در جذب مستعمرات امپراتوری‌های رقیب، و به سیطره در آوردن آنها وابسته خواهد بود. در نتیجه، در جریان رقابت‌های امپریالیستی، به تدریج بر قدرت امپراتوری‌های بزرگتر افزوده شده و امپراتوری‌های ضعیف‌تر، حذف می‌شود. امپراتوری‌ها برای افزایش قدرت خود، مجبور خواهند شد تا مستعمرات خود را نیز پیشرفت دهند(توکلی مقدم و همکاران،۱۳۹۲). در ادامه هریک از الگوریتم رقابت استعماری توضیح داده خواهدشد.

۲-۵-٤-۱ . .ایجاد جواب‌های اولیه (شکل دهی امپراتوری‌های اولیه)

هدف در بهینه‌سازی، یافتن یک جواب بهینه بر حسب متغیرهای مسأله است. به‌همین منظور لازم است ابتدا مجموعه‌ای از جواب اولیه ایجاد شود. در الگوریتم ژنتیک، به این جواب‌ها، کروموزوم[1] می‌گویند. در ICA .نیز به جواب‌ها یک کشور می‌گویند. در یک مسأله‌ی بهینه‌سازی N_{var} بعدی، یک کشور، یک آرایه‌ی $1 \times N_{var}$ است. این آرایه به صورت زیر تعریف می‌شود.

$$country = [p_1, p_2, p_3, \cdots, p_{N_{var}}] \qquad (۷۱-۲)$$

مقادیر متغیرها در یک کشور، به صورت اعداد اعشاری نمایش داده می‌شوند. از دیدگاه تاریخی – فرهنگی، اجزاءی تشکیل دهنده یک کشور را می‌توان ویژگی های اجتماعی– سیاسی آن کشور، همچون فرهنگ، زبان، ساختار اقتصادی و سایر ویژگی‌ها در نظر گرفت. این مساله در شکل ۲-۱٤(اجزای اجتماعی سیاسی تشکیل دهنده یک کشور) به خوبی نشان داده شده است. مطابق این شکل متغیرهای مجهول تابع هزینه، که مقدار آن طی فرایند بهینه‌سازی مشخص می‌شود، در نگاه اجتماعی ـ سیاسی ویژگی‌های تاریخی و فرهنگی‌ای هستند که یک کشور را به نقطه کمینه تابع هزینه هدایت می‌کنند. در حقیقت در حل یک مسأله بهینه‌سازی توسط الگوریتم معرفی شده، یافتن بهترین کشور (کشوری با بهترین ویژگی های اجتماعی ـ سیاسی) هدف اصلی مساله است. یافتن این کشور معادل یافتن بهترین پارامترهای مسأله است که کمترین مقدار تابع هزینه را ایجاد می‌کنند.



شکل ۲-۱۴(اجزای اجتماعی سیاسی تشکیل دهنده یک کشور(توکلی مقدم و همکاران،۱۳۹۲))

فرض کنید می‌خواهیم یک کنترل کننده PID برای یک سیستم کنترلی، با کمترین میزان مجموع فراجهش و انتگرال قدر مطلق خطا، طراحی کنیم. برای حل این مسأله مجموعه‌ای از جواب‌های ممکن را به صورت جواب-های اولیه ایجاد می‌کنیم. بنابراین کشور i–اُم به صورت رابطه (۲-۷۲) تعریف می‌شود.

[1] chromosome

$$country_i = [KP_i, KI_i, KD_i] \quad (۷۲-۲)$$

برای شروع الگوریتم باید تعدادی از این کشورها (به تعداد کشورهای اولیه الگـوریتم) ایجـاد شـوند. بنـابراین ماتریس کل کشورها به صورت تصادفی اولیه تشکیل می‌شود.

$$COUNTRY = \begin{bmatrix} country_1 \\ country_2 \\ country_3 \\ \vdots \\ country_{N_{country}} \end{bmatrix} = \begin{bmatrix} KP_1 & KI_1 & KD_1 \\ KP_2 & KI_2 & KD_2 \\ \vdots & \vdots & \vdots \\ KP_{N_{country}} & KI_{N_{country}} & KD_{N_{country}} \end{bmatrix} \quad (۷۳-۲)$$

هزینه‌ی یک کشور با ارزیابی تابع f در متغیرهای $[p_1, p_2, p_3, \cdots, p_{N_{var}}]$ یافته می‌شود. بنابراین:

$$cost_i = f(country_i) = f(p_1, p_2, p_3, \dots, p_{N_{var}}) \quad (۷٤-۲)$$

در مسأله طراحی کنترل کننده، با هدف نظر، این تابع به صورت رابطه (۲-۷۵)خواهد بود.

$$F = w1 \times MaxOvershoot + w2 \times IAE \quad (۷۵-۲)$$

که در آن MaxOvershoot بیشینه فراجهش و IAE انتگرال قدر مطلـق خطـا اسـت. w1 و w2 نیـز وزن‌هـایی هستند که میزان اهمیت هر یک از هدف‌ها را نشان می‌دهند. بنابراین، کاری کـه بـرای بدسـت آوردن هزینـه یـک کشور (دسته پارامترهای کنترل کننده PID) باید انجام شود، این است که هر دسته از این ضرایب به عنـوان کنتـرل کننده در نظر گرفته شوند.در نهایت مجموع بیشـینه فـراجهش و انتگـرال قـدرمطلق خطـا، بـه‌عنـوان هزینـه ایـن کشور(ضرایب کنترل کننده) محاسبه می‌شوند. هدف بدست آوردن بهترین کشـور (بهتـرین دسـته ضـرایب کنتـرل کننده) است. الگوریتم رقابت استعماری، با تولید مجموعه‌ای اولیه از این ضـرایب و دسـته بنـدی آنهـا در قالـب امپراتوری‌ها و اعمال سیاست جذب از طـرف اسـتعمارگران بـه روی مسـتعمره‌هـا و همچنـین بـا ایجـاد رقابـت استعماری میان امپراتوری‌ها، به جست‌وجوی بهترین کشور می‌پردازد.

برای شروع الگوریتم، تعداد $N_{country}$ کشور اولیه را ایجاد می‌کنـیم. N_{imp} تـا از بهتـرین اعضـای ایـن جمعیـت (کشورهای دارای کمترین مقدار تابع هزینه) را به عنوان امپریالیست برمی‌گزینیم. باقیمانده N_{col} کشور، مسـتعمراتی را تشکیل می‌دهند که هرکدام به یک امپراتوری تعلق دارند. برای تقسیم مستعمرات اولیه بین امپریالست‌ها، به هـر امپریالیست‌ها، تعدادی از مستعمره‌ها را متناسب با قدرت هر امپریالیست، به آن تخصیص می‌دهیم. انجام این کـار، با داشتن هزینه همه امپریالیست‌ها امکان‌پذیر است. به‌همین منظـور ابتـدا هزینـه نرمـالیزه آنهـا را براسـاس رابطـه (۲-۷٦)به‌دست می‌آوریم.

$$C_n = \max_i\{c_i\} - c_n \quad (۷٦-۲)$$

که در آن C_n، هزینه امپریالست n–اُم، $\max_i\{c_i\}$ بیشترین هزینه میان امپریالیست‌ها و C_n، هزینه نرمالیزه شده این امپریالیست است. هر امپریالیست با هزینه بیشتر، یعنی امپریالیست ضعیف‌تر، هزینه نرمالیزه کمتری خواهد داشت، با داشتن هزینه نرمالیزه، قدرت نسبی نرمالیزه‌ی هر امپریالیست، به صورت رابطه (۲-۷۷) محاسبه و برمبنای آن، کشورهای مستعمره، بین امپریالسیت‌ها تقسیم می‌شوند.

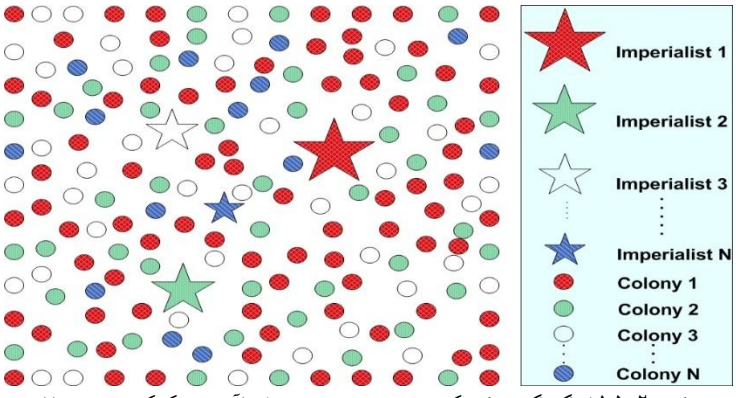
$$p_n = \left| \frac{c_n}{\sum_{i=1}^{N_{var}} c_i} \right| \qquad (۲-۷۷)$$

به عبارت دیگر، قدرت نرمالیزه شده یک امپریالیست، نسبت مستعمراتی است که توسط آن امپریالیست اداره می‌شود. بنابراین تعداد اولیه‌ی مستعمرات یک امپریالیست برابر خواهد بود با:

$$N.C_n = round\{p_n.(N_{col})\} \qquad (۲-۷۸)$$

که در آن $N.C_n$، تعداد اولیه مستعمرات یک امپراتوری و N_{col} نیز تعداد کل کشورهای مستعمره موجود در جمعیت کشورهای اولیه است. $round$ نیز تابعی است که نزدیک‌ترین عدد صحیح به یک عدد اعشاری را می‌دهد. با در نظر گرفتن $N.C$ برای هر امپراتوری، این تعداد از کشورهای مستعمره اولیه را به طور تصادفی انتخاب کرده، به امپریالیست n–اُم می‌دهیم. با داشتن حالت اولیه تمامی امپراتوری‌ها، الگوریتم رقابت استعماری شروع می‌شود. روند تکامل در یک حلقه قرار دارد که تا برآورده شدن یک شرط توقف، ادامه می‌یابد.

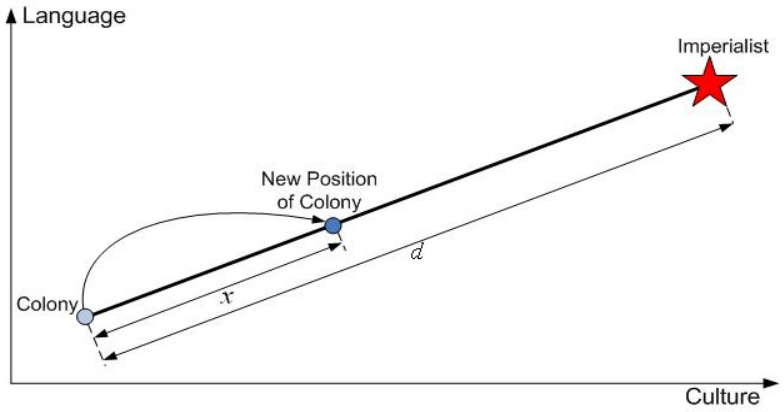
شکل ۲-۱۵(چگونگی شکل‌گیری امپراتوری‌های اولیه) چگونگی شکل‌گیری امپراتوری‌های اولیه را نشان می‌دهد. همانگونه که در این شکل نشان داده شده است. امپراتوری‌های بزرگتر، مستعمره بیشتری دارند. در این شکل، امپریالست اول قوی‌ترین امپراتوری را ایجاد کرده است و بیش‌ترین تعداد مستعمره را دارد(توکلی مقدم و همکاران،۱۳۹۲).



شکل ۲-۱۵(چگونگی شکل‌گیری امپراتوری‌های اولیه(آتش پزگرگری، ۱۳۸۷))

۲-۵-٤-۲ . مدل‌سازی سیاست جذب(حرکت مستعمره‌ها به سمت امپریالیست)

سیاست همگون‌سازی[1] (جذب) با هدف تحلیل فرهنگ و ساختار اجتماعی مستعمرات در فرهنگ حکومت مرکزی انجام می‌گرفت. همان‌گونه که پیش‌تر نیز بیان شد، کشورهای استعمارگر برای افزایش نفوذ خود شروع به ایجاد عمران از جمله زیرساخت‌های حمل و نقل، تاسیس دانشگاه و غیره کردند. کشورهایی نظیر انگلیس و فرانسه با درپیش گرفتن سیاست همگون‌سازی، در مستعمرات خود در فکر ایجاد انگلیس نو[2] و فرانسه نو[3] در مستعمرات خویش بودند. با در نظر گرفتن شیوه نمایش یک کشور در حل مسئله بهینه‌سازی، در حقیقت این حکومت مرکزی با اعمال سیاست جذب سعی داشت تا کشور مستعمره را در ابعاد مختلف اجتماعی – سیاسی به خود نزدیک کند. این بخش از فرایند استعمار در الگوریتم بهینه‌سازی، به صورت حرکت مستعمرات به سمت کشور امپریالیست، مدل شده است. شمای کلی این حرکت در شکل ۲-۱٦ نشان داده شده است.



شکل ۲-۱٦(شمای کلی حرکت مستعمرات به سمت امپریالیست(آتش پزگرگری، ۱۳۸۷))

مطابق این شکل کشور امپریالیست کشور مستعمره را در راستای محورهای فرهنگ و زبان به سمت خود جذب می‌کند. همان‌گونه که در این شکل ۲-۱٦(شمای کلی حرکت مستعمرات به سمت امپریالیست(آتش پزگرگری، ۱۳۸۷)) نشان داده شده است، کشور مستعمره[4]، به اندازه x واحد در جهت خط واصل مستعمره به استعمارگر[5]، حرکت کرده و به موقعیت جدید[6]، کشانده می‌شود. در این شکل، فاصله میان استعمارگر و مستعمره با

---

[1] Assimilation
[2] New England
[3] New France
4 Colony
5 Imperialist
6 New Position of Colony

d نشان داده شده است. x نیز عددی تصادفی با توزیع یکنواخت (و یا هر توزیع مناسب دیگر) می‌باشد. یعنی برای x داریم:

$$x \in \cup\,(0\,,\beta \times d) \qquad (۷۹-۲)$$

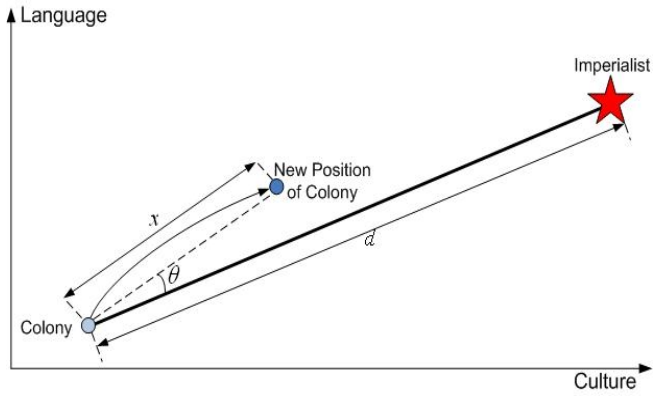
که در رابطه (۲-۷۹)، β عددی بزرگتر از یک و نزدیک به ۲ است.β=۲ می‌تواند یک انتخاب مناسب باشد. وجود ضریب $\beta > 1$ باعث می‌شود کشور مستعمره در حین حرکت به سمت کشور استعمارگر، از جهت‌های مختلف به آن نزدیک شود.

با بررسی تاریخی پدیده همگون‌سازی، این حقیقت آشکار می‌شود که هر چند کشورهای استعمارگر بطور جدی پیگیر سیاست جذب بوده‌اند، اما وقایع بطور کامل مطابق سیاست اعمال شده آنها پیش نمی‌رفت و انحرافاتی در نتیجه کار وجود داشت. در الگوریتم معرفی شده، این انحراف احتمالی با افزودن یک زاویه تصادفی به مسیر جذب مستعمرات، انجام می‌گیرد. بدین منظور، در حرکت مستعمرات به سمت استعمارگر، کمی زاویه تصادفی نیز به جهت حرکت مستعمره، اضافه می‌شود.

این حالت در شکل ۲-۱۷ (حرکت واقعی مستعمرات به سمت امپریالیست) نشان داده شده است. بدین منظور این بار به‌جای حرکت به اندازه x به سمت کشور استعمارگر و در جهت بردار واصل مستعمره به استعمارگر، به همان میزان، ولی با انحراف θ در مسیر، به حرکت خود ادامه می‌دهیم. θ را به صورت تصادفی و با توزیع یکنواخت در نظر می‌گیریم (اما هر توزیع دلخواه و مناسب دیگر نیز می‌تواند استفاده شود).

$$\theta \in \cup\,(-\gamma,\gamma) \qquad (۸۰-۲)$$

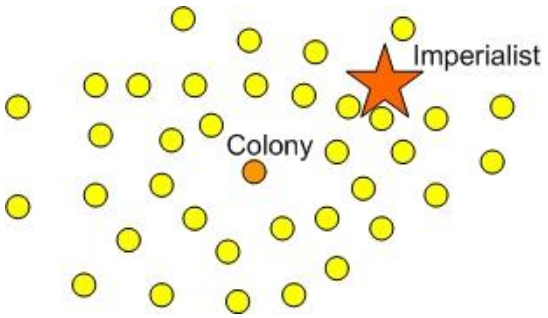
در این رابطه، γ پارامتری دلخواه است که افزایش آن باعث افزایش جستجوی اطراف امپریالیست می‌شود و کاهش آن نیز موجب می‌شود مستعمرات، تا حد ممکن، به بردار واصل مستعمره به استعمارگر، نزدیک حرکت کنند. با در نظر گرفتن واحد رادیان برای θ ، عددی نزدیک به $\pi/4$، در اکثر پیاده‌سازی ها، انتخاب مناسبی بوده است(توکلی مقدم و همکاران،۱۳۹۲).



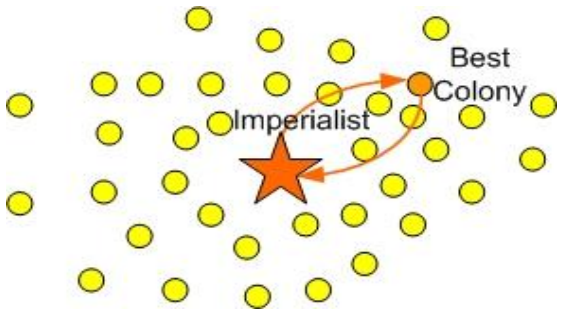
شکل ۲-۱۷ (حرکت واقعی مستعمرات به سمت امپریالیست(آتش پزگرگری، ۱۳۸۷))

۲-۵-٤-۳ . جابه‌جایی موقعیت مستعمره و امپریالیست

سیاست جذب، در عین نابودی ساختارهای اجتماعی – سیاسی کشور مستعمره، در بعضی موارد نتایج مثبتی را نیز برای آنها در پی داشت. بعضی از کشورها در نتیجه اعمال این سیاست به نوعی از خودباوری عمومی دست یافتند و پس از مدتی همان تحصیل‌کردگان (به عبارت دیگر جذب شدگان فرهنگ استعماری) بودند که به رهبری ملت خود برای رهایی از چنگال استعمار پرداختند. نمونه های فراوانی از این موارد را می‌توان در مستعمرات انگلیس و فرانسه یافت. از سوی دیگر، نگاهی به فرازونشیب چرخش قدرت در کشورها به خوبی نشان می‌دهد که کشور هایی که زمانی در اوج قدرت سیاسی – نظامی بودند، پس از مدتی سقوط کردند و در مقابل، کشورهایی سکان قدرت را در دست گرفتند که زمانی هیچ قدرتی نداشتند. در مدل سازی این واقعه تاریخی در الگوریتم معرفی شده، به این صورت عمل شده است که در حین حرکت مستعمرات به سمت کشور استعمارگر، ممکن است بعضی از این مستعمرات به موقعیتی بهتر از امپریالیست برسند؛ یعنی به نقاطی در تابع هزینه برسند که هزینه کمتری را نسبت به مقدار تابع هزینه در موقعیت امپریالیست، تولید می‌کنند. در این حالت، کشور استعمارگر و کشور مستعمره، جای خود را با همدیگر عوض کرده، الگوریتم با کشور استعمارگر در موقعیت جدید ادامه می‌یابد و این بار این کشور امپریالیست جدید است که شروع به اعمال سیاست همگون‌سازی بر مستعمرات خود می‌کند. تغییر جای استعمارگر و مستعمره، در شکل ۲-۱۸(تغییر جای استعمارگر و مستعمره) نشان داده شده است. در این شکل، بهترین مستعمره‌ی امپراتوری، که هزینه‌ای کمتر از خود امپریالیست دارد، به رنگ تیره‌تر، نشان داده شده است. شکل ۲-۱۹(کل امپراتوری، پس از تغییر موقعیت‌ها، کل امپراتوری را پس از تغییر موقعیت‌ها، نشان می‌دهد. (توکلی مقدم و همکاران،۱۳۹۲).



شکل ۲-۱۸(تغییر جای استعمارگر و مستعمره(آتش پزگرگری، ((۱۳۸۷



شکل ۲-۱۹(کل امپراتوری، پس از تغییر موقعیت‌ها(آتش پزگرگری، ((۱۳۸۷

۲-۵-٤-٤ . قدرت کل یک امپراتوری

قدرت یک امپراتوری برابر است با قدرت کشور استعمارگر، که درصدی از قدرت کل مستعمرات امپراتور نیز به آن اضافه می‌شود. بدین ترتیب هزینه کل یک امپراتوری براساس رابطه (۲-۸۱) محاسبه می‌شود.

$$TC_n = cost(imperialist_n) + \xi\ mean\{cost(colonies\ of\ empire_n)\} \qquad (۸۱-۲)$$

که در آن TC_n هزینه کل امپراتوری n اُم و ξ عددی مثبت است که معمولاً بین صفر و یک و نزدیک به صفر در نظر گرفته می‌شود. کوچک در نظر گرفتن ξ، باعث می‌شود که هزینه کل یک امپراتوری، تقریباً برابر با هزینه حکومت مرکزی آن (کشور امپریالیست)، شود و افزایش ξ، نیز منجر به افزایش تاثیر میزان هزینه مستعمرات یک امپراتوری در تعیین هزینه کل آن می‌شود. در حالت کلی $\xi = 0.05$ در اکثر پیاده‌سازی به جواب‌های مطلوبی منجر شده است.

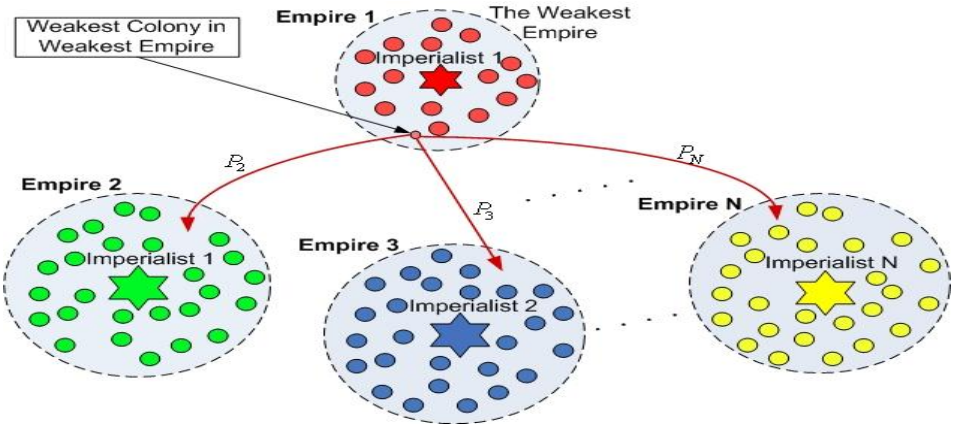
۲-۵-٤-۵ . .رقابت استعماری

همان‌طور که قبلاً نیز بیان شد، هر امپراتوری‌ای که نتواند بر قدرت خود بیفزاید و قدرت رقابت خود را از دست بدهد، در جریان رقابت‌های امپریالیستی، حذف خواهد شد. بدین معنی که به مرور زمان، امپراتوری‌های ضعیف، مستعمرات خود را از دست می‌دهند و امپراتوری‌های قوی‌تر، این مستعمرات را تصاحب کرده، بر قدرت خویش می‌افزایند. برای مدل کردن این واقعیت، فرض می‌کنیم که امپراتوری در حال حذف، ضعیف‌ترین امپراتوری موجود است. بدین ترتیب، در تکرار الگوریتم، یک یا چند تا از ضعیف‌ترین مستعمراتِ ضعیف‌ترین امپراتوری را برداشته، برای تصاحب این مستعمرات، رقابتی را میان همه امپراتوری‌ها ایجاد می‌کنیم. مستعمرات مذکور، لزوماً توسط قویترین امپراتوری، تصاحب نخواهند شد، بلکه امپراتوری‌های قویتر، احتمال تصاحب بیشتری دارند. شکل ۲-۲۰ (شمای کلی رقابت استعماری: امپراتوری‌های بزرگ‌تر، با احتمال بیشتر، مستعمرات

امپراتوری‌های دیگر را تصاحب می‌کنند(آتش پزگرگری، ۱۳۸۷) عملکرد کلی این بخش از الگوریتم را نشان می‌دهد.

در شکل۲-۲۰ امپراتوری شماره ۱ به عنوان ضعیف‌ترین امپراتوری در نظر گرفته شده و یکی از مستعمرات آن در معرض رقابت امپریالیستی قرار گرفته است و امپراتوری‌های ۲ تا N برای تصاحب آن با هم رقابت می‌کنند. برای مدل‌سازی رقابت میان امپراتوری‌ها برای تصاحب این مستعمرات، ابتدا احتمال تصاحب هر امپراتوری، به تناسب قدرت آن امپراتوری و با در نظر گرفتن هزینه کل امپراتوری، به ترتیب رابطه(۲-۸۲)محاسبه می‌شود. آنگاه از روی هزینه کل امپراتوری، هزینه کل نرمالیزه شده آن تعیین می‌شود.

$$N.T.C_n = \max_i\{T.C_i\} - T.C_n \qquad (۲-۸۲)$$



شکل ۲-۲۰ (شمای کلی رقابت استعماری: امپراتوری‌های بزرگ‌تر، با احتمال بیشتر، مستعمرات امپراتوری‌های دیگر را تصاحب می‌کنند(آتش پزگرگری، ۱۳۸۷))

در این رابطه $T.C_n$، هزینه کل امپراتوری n-اُم و $N.T.C_n$ نیز، هزینه کل نرمالیزه شده آن امپراتوری است. هر امپراتوری که $T.C_n$ کمتری داشته باشد $N.T.C_n$بیشتری خواهد داشت. در حقیقت$T.C_n$ معادل هزینه کل یک امپراتوری $N.T.C_n$ و معادل قدرت کل آن است. امپراتوری با کمترین هزینه، دارای بیشترین قدرت است. با داشتن هزینه کل نرمالیزه شده، احتمال (قدرت) تصاحب مستعمره رقابت، توسط هر امپراتوری، به صورت (۲-۸۳)محاسبه می‌شود.

$$p_{p_n} = \left| \frac{N.T.C_{.n}}{\sum_{i=1}^{N_{imp}} N.T.C_{.i}} \right| \qquad (۲-۸۳)$$

با داشتن احتمال تصاحب هر امپراتوری، مکانیزمی همانند چرخه رولت[1] در الگوریتم ژنتیک مورد نیاز است تا مستعمره مورد رقابت را با احتمال متناسب با قدرت امپراتوری‌ها در اختیار یکی از آنها قرار دهد. در کنار امکان استفاده از چرخ رولت موجود، در این نوشتار مکانیزم جدیدی برای پیاده‌سازی این فرایند معرفی شده است که نسبت به چرخه رولت هزینه محاسباتی بسیار کم‌تری دارد. زیرا عملیات نسبتاً زیاد مربوط به محاسبه تابع توزیع جمعی احتمال(CDF)[2] را که در چرخه رولت مورد نیاز است را حذف می‌کند و فقط به داشتن تابع چگالی احتمال(PDF)[3] نیاز دارد. در ادامه مکانیزم مطرح شده برای اختصاص متناسب با احتمال مستعمره مورد رقابت به امپراتوری های رقیب توضیح داده می‌شود.

با داشتن احتمال تصاحب هر امپراتوری، برای اینکه مستعمرات مذکور را به صورت تصادفی، ولی با احتمال وابسته به احتمال تصاحب هر امپراتوری، بین امپراتوری‌ها تقسیم کنیم؛ بردار P را از روی مقادیر احتمال فوق، به صورت رابطه (۲-۸٤)شکل می‌دهیم.

$$P = \left[p_{p_1}, p_{p_2}, p_{p_3}, \dots, p_{p_{N_{imp}}}\right] \quad (۲-۸٤)$$

بردار P دارای اندازه $1\times N_{imp}$ است و از مقادیر احتمال تصاحب امپراتوری‌ها تشکیل شده است. سپس بردار تصادفی R، هم‌اندازه با بردار R تشکیل شود. آرایه‌های این بردار، اعدادی تصادفی با توزیع یکنواخت در بازه $[0,1]$ است.

$$R = \left[r_1, r_2, r_3, \cdots, r_{N_{imp}}\right] \quad (۲-۸۵)$$

به‌طوری‌که:

$$r_1, r_2, r_3, \cdots, r_{N_{imp}} \in U(0,1) \quad (۲-۸٦)$$

سپس بردار Dرا به صورت زیر رابطه(۲-۸۷) تشکیل می‌شود.

$$\begin{aligned} D = P - R &= \left[D_1, D_2, D_3, \cdots, D_{N_{imp}}\right] \\ &= \left[p_{p_1} - r_1, p_{p_2} - r_2, p_{p_3} - r_3, \cdots, p_{p_{N_{imp}}} - r_{N_{imp}}\right] \end{aligned} \quad (۲-۸۷)$$

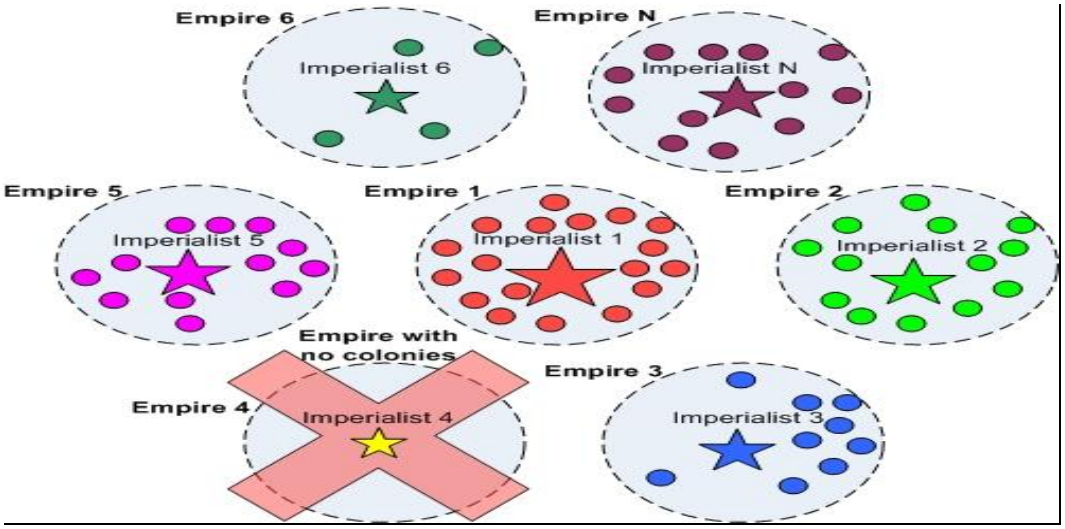
هر امپراطوری که بیشترین احتمال تصاحب را داشته باشد، اندیس مربوط به آن در بردار D، با احتمال بیشتری بالاترین مقدار را خواهد داشت. عدم نیاز به محاسبه CDF باعث می‌شود که این مکانیزم نسبت به چرخه رولت

[1] Roulette Wheel
[2] Cumulative Distribution Function (CDF)
[3] Probability Density Function (PDF)

با سرعت به مراتب بیشتری عمل کند. مکانیزم جدید مطرح شده نه تنها می‌تواند در اختصاص مستعمره به امپراتوری بر حسب احتمال تصاحب آنها مفید باشد، بلکه به عنوان یک مکانیزم انتخاب بر حسب احتمال می‌تواند جایگزین چرخه رولت در الگوریتم ژنتیک برای انتخاب والدین شود و سرعت اجرای عملیات در آن را تا حد زیادی افزایش دهد. با تصاحب مستعمره توسط یکی از امپراتوری ها، عملیات این مرحله از الگوریتم نیز به پایان می‌رسد(توکلی مقدم و همکاران،۱۳۹۲).

۲-۵-٤-٦ . سقوط امپراتوری‌های ضعیف

همان‌طور که بیان شد، در جریان رقابت‌های امپریالیستی، در طول زمان، امپراتوری‌های ضعیف به تدریج سقوط کرده، مستعمراتشان به دست امپراتوری‌های قوی‌تر می‌افتد. شرط‌های متفاوتی را می‌توان برای سقوط یک امپراتوری در نظر گرفت. در الگوریتم پیشنهاد شده، یک امپراتوری زمانی حذف شده تلقی می‌شود که مستعمرات خود را از دست داده باشد. این مسأله در شکل ۲-۲۱ (سقوط امپراتوری ضعیف؛ امپراتوری شماره ٤، به علت از دست دادن کلیه مستعمراتش، دیگر قدرتی برای رقابت ندارد و باید از میان بقیه امپراتوری‌ها حذف شود(آتش پزگرگری، ۱۳۸۷) به‌خوبی نشان داده شده است. در این شکل، امپراتوری چهارم به علت از دست دادن همه مستعمراتش، دیگر قدرتی برای رقابت ندارد و باید از میان بقیه امپراتوری‌ها حذف شود(توکلی مقدم و همکاران،۱۳۹۲).



شکل ۲-۲۱ (سقوط امپراتوری ضعیف؛ امپراتوری شماره ٤، به علت از دست دادن کلیه مستعمراتش، دیگر قدرتی برای رقابت ندارد و باید از میان بقیه امپراتوری‌ها حذف شود(آتش پزگرگری، ۱۳۸۷))

۲-۵-٤-۷ . همگرایی

الگوریتم مورد نظر، تا برآورده شدن یک شرط همگرایی، و یا تا اتمام تعداد کل تکرارها، ادامه می‌یابد. پس از مدتی، همه امپراتوری‌ها، سقوط کرده و تنها یک امپراتوری خواهیم داشت. بقیه کشورها تحت کنترل این امپراتوری واحد قرار می‌گیرند. در این دنیای ایده‌آل جدید، همه‌ی مستعمرات را یک امپراتوری واحد اداره می‌کند و موقعیت‌ها و هزینه‌های مستعمرات، برابر با موقعیت و هزینه کشور امپریالیست است. در دنیای جدید، نه تنها میان مستعمرات، بلکه میان مستعمرات و کشور امپریالیست، تفاوتی وجود ندارد. به عبارت دیگر، همه‌ی کشورها در عین حال، هم مستعمره و هم استعمارگرند. در چنین موقعیتی رقابت امپریالیستی به پایان رسیده و به عنوان یکی از شروط توقف الگوریتم متوقف می‌شود(توکلی مقدم و همکاران،۱۳۹۲).

۲-۵-۵ . کاربردها

الگوریتم معرفی شده به‌عنوان نسخه ابتدایی یک الگوریتم مبتنی بر یک فرآیند اجتماعی- سیاسی و به‌طور ویژه پدیده پیچیده استعمار است. بنابراین، می‌توان اصلاحاتی نیز در آن ایجاد کرد. الگوریتم معرفی شده در حال حاضر برای حل مسائل گسسته بهینه‌ساری مناسب است. برای حل مسائل گسسته بهینه‌سازی باید تغییراتی در الگوریتم اعمال شود.

ارائه نسخه گسسته الگوریتم می‌تواند برای حل مسائلی همچون انتخاب ورودی در سیستم‌ها و انتخاب ویژگی برای اهداف بازشناسی الگو مفید باشد. الگوریتم‌های رایجی مانند بهینه‌سازی گروه ازدحام نیز در نسخه ابتدایی خود برای حل مسائل پیوسته مطرح شده بودند و بعدها نسخه‌های گسسته آنها معرفی شده است.

در کاربردهای اعمال شده نیز، همه مسائل بهینه سازی فقط یک تابع هدف داشتند. الگوریتم مطرح شده کنونی می‌تواند برای حل مسائل بهینه سازی چند هدفه[1] و برای یافتن جبهه پارتو[2] نیز استفاده شود ولی نتایج بدست آمده از آن به خوبی نتایج الگوریتم‌های بهینه سازی مخصوص مسائل چند هدفه (همانند NSGA-II، نسخه چند هدفه الگوریتم ژنتیک) نخواهد بود.

جدول ۲-۳ (کاربردهای الگوریتم رقابت استعماری(توکلی مقدم و همکاران،۱۳۹۲))

نوع کاربرد	مراجع

[1] Multi-Objective Optimization
2 Pareto Front

مهندسی برق- قدرت	Rabiee, A.and others:2011; Rashtchi, V. and others:2011; Kohansal, M.and others: 2012.
مهندسی کنترل	Jalalzadeh, S.and others: 2011; Rismanbor, M. and Karim faez: 2012.
پردازش تصویر	Razmjooy, N. and others: 2011; Samaneh K.and S.B shokouhi: 2012.
زنجیره تامین	Naimi Sadigh, A and others: 2012.
زمان‌بندی و طراحی خط تولید	Lian, K. and others: 2012.
مهندسی عمران	Bagheri, A.and G.G. Amiri: 2011.
داده کاوی	Khademolghorani, F.: 2011
پردازش سیگنال	Sheikhan, M. and others: 2011.
زمان‌بندی	Chen, R. and others: 2012; Ayough, A.:2011.
زنجیره تامین	Kaviani, M. and others: 2011; Naimi Sadigh, A. and others: 2011.
مسأله فروشنده دوره‌گرد	Nemati, K. and others: 2011.
شبکه‌های عصبی	Ahmadi, M. A.: 2011; Khorani, V. and others: 2011; Abdechiri, M. and others: 2010; Tayefeh Mahmoudi, M. and others: 2009.
طراحی سیستم انتقال حرارت	Karami, A. and others: 2011; Rezaei, E. and others: 2012; Yousefi, M. and others: 2011.
مدیریت بحران	Moadi, S. and others: 2011.
شبکه‌های توزیع	Chahkandi Nejad, H. and others: 2011.
برنامه‌ریزی فرایند	Lian, K.: 2011.

در ادامه کار می‌توان با اعمال تغییراتی در ساختار الگوریتم، آن را برای حل مسائل بهینه‌سازی چندهدفه نیز مناسب سازی کرد. همچنین، الگوریتم توسعه داده شده، در وهله اول با نگاهی کاملا نو به مبحث بهینه‌سازی، پیوندی جدید میان علوم انسانی و اجتماعی از یک سو و علوم فنی و ریاضی از سوی دیگر، برقرار می‌کند. ارتباط میان این دو شاخه از علم بگونه‌ای است که غالبا ریاضیات به‌عنوان ابزاری دقیق و قوی در خدمت علوم انسانی کلی‌نگر قرار گرفته، به درک و تحلیل نتایج آن کمک می‌کند. اما الگوریتم توسعه داده شده بر خلاف معمول، نقطه قوت علوم انسانی و اجتماعی، یعنی کلی‌نگری و وسعت دید آن را به خدمت ریاضیات درآورده، از آن به‌عنوان

ابزاری برای درک بهتر ریاضیات و حل بهتر مسائل ریاضی استفاده می‌کند. بنابراین حتی بدون درنظر گرفتن قابلیت‌های ریاضی و علمی روش توسعه داده شده، پیوند ایجاد شده میان این دو شاخه به ظاهر جدا از هم، به عنوان یک پژوهش میان رشته‌ای، در نوع خود دارای ارزش بسیاری است. به‌طور کلی مزایای الگوریتم اجتماعی پیشنهادی را می‌توان به‌صورت زیر خلاصه کرد:

- نو بودن ایده پایه‌ای الگوریتم، به‌عنوان اولین الگوریتم بهینه‌سازی مبتنی بر فرایند اجتماعی – سیاسی.
- توانایی بهینه‌سازی هم‌تراز و حتی بالاتر در مقایسه با الگوریتم‌های مختلف بهینه‌سازی، در مواجهه با انواع مسائل بهینه‌سازی.
- سرعت مناسب یافتن جواب بهینه.

همانطور که بیان شد، روش‌های تکاملی، ویژگی گریز از نقطه کمینه محلی را دارد. در مقابل روش‌های کلاسیک بهینه سازی دارای سرعت همگرایی بیشتری هستند. برای داشتن سرعت همگرایی بالا و در عین حال نیفتادن در نقاط بهینه محلی، یک روش رایج، ترکیب الگوریتم‌های تکاملی با روش‌های کلاسیک بهینه‌سازی همچون روش نیوتون است. در ادامه کار می‌توان ترکیبی از الگوریتم مطرح شده را نیز با الگوریتم‌های کلاسیک بهینه‌سازی ترکیب کرد. برخی از کاربردهای الگوریتم رقابت استعماری را می‌توانید درجدول ۲-۳ (کاربردهای الگوریتم رقابت استعماری(توکلی مقدم و همکاران،۱۳۹۲)) مشاهده کنید(توکلی مقدم و همکاران،۱۳۹۲).

۲-٦. پیشینه تحقیق

۲-٦-۱ . پیشینه داخلی

مقصود امیری در رساله‌ی دکتری خود با عنوان «توسعه سیستم‌های پایایی با اجزاء تعمیر پذیر» و با راهنمایی دکتر فرهاد قاسمی طاری در دانشگاه شریف به معرفی انواع سیستم‌های تعمیر پذیر به شرح زیر پرداخته است.

۱. سیستم‌های با اجزاء تعمیر پذیر بصورت سری
۲. سیستم‌های با اجزاء تعمیر پذیر بصورت موازی
۳. سیستم‌های با اجزاء تعمیر پذیر بصورت k عنصر از n عنصر
٤. سیستم‌های با اجزاء تعمیر پذیر بصورت آماده بکار

۵. سیستم‌های با اجزاء تعمیر پذیر بصورت تقسیم بار
٦. سیستم‌های پایایی و صف با پارامترهای فازی
۷. سایر سیستم‌های پایایی

و همچنین با استفاده از مدل‌های مارکف، مقادیر ویژه و بردار ویژه، روشی برای تحلیل گذرای پایایی، دسترس پذیری و ماندگاری سیستم های ذکر شده ارائه داده است.

اسمائیل آتش پز گرگری دانشجوی کارشناسی ارشد دانشکده مهندسی برق و کامپیوتر دانشگاه تهران در سال ۱۳۸۷ در پایان نامه خود به راهنمایی دکتر کارلوس تحت عنوان «توسعه الگوریتم بهینه سازی اجتماعی و بررسی کارآیی آن» به ابداع الگوریتم رقابت استعماری دست یافت و کاربردها و کاربردهای این الگوریتم را با ارائه مثال-هایی نشان داد.

عایشه کرد دانشجوی کارشناسی ارشد در رشته علوم کامپیوتر گرایش محاسبات علمی در سال ۱۳۹۰در پایان نامه خود به راهنمایی دکتر رسول رمضانیان تحت عنوان «انتخاب و بهینه سازی سبد سهام با استفاده از روش‌های فرا ابتکاری از جمله الگوریتم رقابت استعماری به بهینه‌سازی سبد سهام پرداخته است.

وحید فارضیان و همکارانش، در دومین کنفرانس مهندسی قابلیت اطمینان در سال ۱۳۹۰ مقاله‌ای تحت عنوان «ارائه مدلی برای بهینه سازی قابلیت اطمینان در سیستم سری–موازی بر مبنای مکانیابی تسهیلات» ارائه دادند که در آن به بررسی تمام معیارهای مرتبط با مکان‌یابی به منظور بیشینه نمودن قابلیت اطمینان سیستم سری– موازی در مقابل بروز خرابی‌های ناخواسته و کمینه کردن هزینه مکان‌یابی پرداخته‌اند.

علی اصغر یحی تبار عربی در پایان نامه کارشناسی ارشد خود با راهنمایی دکتر عبدالحمید اشراق جهرمی در سال ۱۳۹۱ در دانشگاه شریف به بررسی سیستم‌های k از n پرداخته و با استفاده از زنجیره مارکف مدل مسأله را بوجود آورده و سپس با استفاده از الگوریتم بهینه سازی انبوه ذرات (pso)[1] به حل مدل پرداخته است.

[1] Particle swarm optimization

۲-٦-۲ . پیشینه خارجی

بدلیل اهمیت زیادی که سیستم‌های k از n در صنعت و در طراحی سیستم دارند این نوع سیستم‌ها توجه بسیاری از مهندسان طراح و محققان را به خود جلب کرده است. در دهه‌های گذشته، مقالات و پژوهش‌های زیادی در مورد دسترس‌پذیری و قابلیت اطمینان سیستم‌های k از n به چاپ رسیده است.

مطالعات فاوزی[۱] وهاوکس[۲] به یک سیستم k از n با اجزای آماده به کار گرم[۳] که با اجزای اضافی آماده به کار سرد[٤] مجهز می‌شود می‌پردازد. زمان‌های تا شکست و تعمیر دارای توزیع نمایی هستند.دسترسی‌پذیری سیستم با یک تعمیرکار و با نرخ خرابی یکسان برای تمامی اجزای در حال کار و آماده به کار بدست می‌آید (Fawzi, et al., 1991). مقدی اس مصطفی[٥] دسترسی سیستم‌های k از n تعمیرپذیر با توزیع طول عمر نمایی و توزیع زمان تعمیر ارلنگ را مورد ارزیابی قرار داد. او در مطالعه‌اش یک سیستم ۲ از ٤ را مورد بررسی قرار داد (Moustafa, et al., 1996). هیژن[٦] و همکارانش به دسترسی‌پذیری یک سیستم k از n که دارای اجزای آماده به کار سرد هستند می‌پردازند با این فرض که تعداد تعمیرکاران برابر تعداد اجزاء است(Heijden, et al., 2001) . مطالعات اخیر ژانگ[۷] و همکارانش روی سیستم k از m+n تعمیرپذیر با اجزاء آماده به کار گرم بوده است که در آن سیستم دارای دو نوع اجزاء است. اجزاء نوع یک دارای m جزء و اجزاء نوع دو دارای n جزء است. اجزاء نوع یک نرخ تعمیر کمتری دارند و چند تعمیر کار مورد لحاظ قرار می‌گیرد. با این وجود این مطالعه هم محدود به سیستم‌های آماده به کار با دو نوع اجزاء می‌شود. برای حل دسترسی پذیری سیستم از رویکرد فرآیند مارکوف استفاده شده است(Zhang, et al., 2005) .

لی[۸] و همکارانش رویکردی برای تحلیل سیستم‌های k از n تعمیرپذیر با اجزاء دارای توزیع طول عمر، نمایی و مستقل از هم ارائه دادند. ارائه آن‌ها با این فرض همراه بود که تعدادی اجزاء در حال کار به کار خود ادامه می‌دهند و در هر زمان که سیستم از کار بیافتد، دوباره بعد از تعمیر سیستم به کار خود ادامه می‌دهد. اما تنها برخی از انواع خرابی‌ها می‌باید رخ دهد تا سیستم بطور کامل از کار بیافتد و قابل تعمیر نباشد(Li, et al., 2006).

---

1 Fawzi
2 Hawkes
3 Warm stand by
4 Cold stand by
5 Maghdi S. Moustafa
6 Der Heijden
7 zhang
8 Li

بارون[1] سیستم k از n با اجزای آماده به کار گرم که با اجزای اضافی آماده به کار سرد مجهز می‌شود را با چند تعمیرکار بررسی می‌کند. فرضیات او شامل توزیع طول عمر اجزاء بصورت نمایی و زمان‌های تعمیر بصورت مرحله‌ای می‌شود(Barron, et al., 2006). فراستیژ [2]و لویکسون[3] با استفاده از فرآیند مارکوف به بررسی یک سیستم k از n می‌پردازد که زمان‌های طول عمر و تعمیر آن از توزیع نمایی پیروی می‌کنند.(Frostig, et al., 2006) .

با توجه به اهمیت مسأله تخصیص سیستم، مطالعات زیادی در این زمینه انجام شده است.

فایف[4] و همکارانش مسأله تخصیص واحدهای اضافی را اولین بار فرمول نویسی کرده است و مسأله را با روش برنامه‌ریزی پویا حل کرده است. در این مسأله تمامی اجزاء فعال فرض شده‌اند (Fyffe, et al., 1968). بولفین[5] و لیو[6] از روش برنامه‌ریزی عدد صحیح برای حل مسأله تخصیص واحدهای اضافی استفاده کرده است(Bulfin, et al.,1985). میسرا[7] با استفاده از روش لاگرانژی به حل مسأله تخصیص افزونگی پرداخته است(Misra, et al., 1991) . کویت[8] و اسمیت[9] با استفاده از روش الگوریتم ژنتیک به حل این دسته از مسائل پرداخته است(Coit, et al.,1996) . کیو[10] و همکارانش (Kuo, et al., 2001). و جن[11] و همکارانش(Gen, et al., 2006) . مطالعه‌ای جامع روی تمامی این روش‌ها انجام داده‌اند و به مقایسه این روش‌ها پرداخته‌اند.

چرن[12] ثابت کرد که حتی یک مسأله تخصیص واحدهای اضافی ساده در سیستم‌های سری با محدودیت‌های خطی یک NP-Hard است(Chern., 1991). این مطلب محققان را برآن داشت تا روش‌های ابتکاری و فرا ابتکاری زیادی برای مسائل تخصیص واحدهای اضافی استفاده کنند. کویت و همکارانش از الگوریتم ژنتیک بطور موفقیت‌آمیزی برای حل مسأله تخصیص واحدهای اضافی استفاده کرده است(Coit, et al.,1996) . سادان[13] و همکارانش با استفاده از روش تابو[14] یک روش کارا برای حل این مسأله ارائه داده است(Sadan, et al., 2003) . چن[1] از الگوریتم ایمیون[2] حل این نوع مسائل استفاده کرده است(Chen., 2006).

---

1 Barron
2 Frostig
3 Levikson
4 Fyffe
5 Bulfin
6 Liu
7 Mitra
8 Coit
9 Srrith
10 Kuo
11 Gen
12 Chern
13 Sedan
14 Tabu

مسأله تخصیص واحدهای اضافی برای ساختارهای مختلفی همچون سری، موازی، شبکه‌ای و همچنین k از n مورد مطالعه قرار می‌گیرد. و نوع دیگر مسائل مورد بررسی در ادبیات سیستم‌های تعمیرپذیر هستند. وقتی صحبت از سیستم‌های تعمیرپذیر به میان می‌آید. یکی از مهم‌ترین شاخص‌های عملکردی آن دسترسی پذیری سیستم است. دیوید[3] و ژیاشن[4] با استفاده از برنامه‌ریزی عدد صحیح به حل مسأله تخصیص واحدهای اضافی با ساختار k از n پرداخته است(David,et al., 2000) . گوپتا[5] و همکارنش به تحلیل عددی دسترسی‌پذیری فرآیندهای سری مورد استفاده در صنعت نفت پرداختند(Gupta, et al., 2005) . کارین[6] و همکارانش دسترسی پذیری یک سیستم k از n را مورد مطالعه قرار دادند که در آن هر کدام از اجزاء تعمیرپذیر و یکسان در نظر گرفته شده‌اند. یک سیاست جایگزینی بلوکی[7] برای نگهداری سیستم مورد استفاده قرار می‌گیرد و تنها یک تعمیرکار و یا تجهیزات تعمیر برای سیستم فرض شده است(Karin, et al., 2007) . شارما[8] و همکارانش با استفاده از الگوریتم ژنتیک، بهینه سازی دسترسی پذیری یک سیستم با ساختار سری- موازی را مورد مطالعه قرار دادند(Sharma, et al., 2008) . ژوانگ[9] و همکارانش با استفاده از سیستم مدیریت دانش به بهینه سازی دسترسی‌پذیری سیستم سری- موازی پرداختند (Juang, et al., 2008). لین مین [10]و همکارانش به مطالعه سیستم سری-موازی با اجزاء تعمیرپذیر و با وابستگی خرابی واحدهای اضافی پرداختند بدین معنی که وابستگی خرابی و محدودیت‌های تیم تعمیر بر روی نرخ انتقال حالت زیر سیستم‌ها اثر گذار است. و در آن به معرفی یک تابع که نرخ خرابی اجزاء را در زیرسیستم‌ها مشخص می‌کند پرداختند و سپس با کمک مدل مارکف و با هدف مینیمم کردن هزینه و تعیین تعداد بهینه واحدهای اضافی مدل مساله را بوجود آوردند. و برای حل مساله بهینه سازی از الگوریتم ژنتیک استفاده کردند (Linmin, et al., 2012).

وینکینگ[11] و همکارانش به آنالیز سیستم تعمیرپذیر k از n با یک تعمیر کار پرداختند که در آن طول عمر هر جزء از توزیع نمایی پیروی می‌کند و هر جزء بعد از تعمیر همانند یک قطعه تازه بکار خود ادامه می‌دهد و با فرضیات ذکر شده به اندازه‌گیری دسترس‌پذیری، نرخ شکست و میانگین زمان خرابی باتکنیک متغیر اضافی و

---

1 Chen
2 Immune Alotithm
3 David
4 Jiachen
5 Gupte
6 Karin
7 Block Replacement Policy
8 Sharma
9 Juang
10 Linmin
11 Wenqing

انتفال لاپلاس پرداختند، و با ارائه مثال عددی به مطالعه تاثیر پارامترهای مختلف بـر روی مقـدار قابلیـت اطمینـان سیستم پرداختند و در نهایت از شبیه سازی مونت کارلو جهت بررسی صحت نتایج بدسـت آمـده اسـتفاده کردنـد (Wenqing, et al., 2014). یوچانـگ[1] و همکـاران یـک روش تحلیلـی بـر پایـه دیـاگرام تصـمیم گیـری چنـد ارزشه(MODS)[2] برای آنالیز قابلیت اطمینان سیستم‌های چند حالته k از n ارائه دادند، و با ارائه نتایج تجربی نشان دادند که روش MOD پیچیدگی محاسباتی کمتری از الگوریتم‌های بازگشتی دارند و می‌تواننـد بطـور مـوثر بـرای حالت‌های واقعی سیستم‌های k از n چند حالته بکار برده شوند(Yuchang, et al., 2015).

در ادامه پیشینه تحقیقات داخلی و خارجی بطور خلاصه در جدول ۲-٤(پیشـینه تحقیقـات داخلـی و خـارجی) آورده شده است.

جدول ۲-٤(پیشینه تحقیقات داخلی و خارجی)

ردیف	عنوان	مراجع
	پیشینه داخلی	
۱	توسعه سیستم‌های پایایی با اجزاء تعمیر پذیر	مقصود امیری،۱۳۸۵
۲	توسعه الگوریتم بهینه سازی اجتماعی و بررسی کارآیی آن	اسمائیل آتش پز گرگری،۱۳۸۷
۳	ارائه مدلی برای بهینـه سـازی قابلیـت اطمینـان در سیسـتم سری–موازی بر مبنای مکانیابی تسهیلات	وحیـــــد فارضـــــیان و همکاران،۱۳۹۰
٤	انتخاب و بهینه سازی سبد سهام با استفاده از روش‌های فرا ابتکاری	عایشه کرد،۱۳۹۰
٥	بهینه سازی دسترس پذیری سیستمهای k از n با الگـوریتم pso	علـــی اصـــغر یحـــی تبـــار عربی،۱۳۹۱
	پیشنه خارجی	
٦	System reliability allocation and a computational algorithm.	(Fyffe, et al., 1968)
۷	Optimal allocation of redundant components for large systems.	(Bulfin, et al.,1985)

1 Yuchang
2 Multi-valued decision diagrams

(Fawzi, et al., 1991)	Availability of an r-out-of-n system with spares and repairs.	٨
(Misra, et al., 1991)	An efficient algorithm to solve integer programming problems arising in system reliability design.	٩
(Chern., 1991).	On the computational complexity of reliability redundancy allocation in a series system.	١٠
(Moustafa, et al., 1996)	Transient analysis of reliability with and without repair for k-out-of-n:G system with two failures modes.	١١
(Coit, et al.,1996)	Reliability optimization of series-parallel systems using a genetic algorithm.	١٢
(David, et al., 2000)	System relibality optimization with k-out-of-n subsystems.	١٣
(Heijden, et al., 2001)	Waiting times at periodically switched one-way traffic lanes.	١٤
(Kuo, et al., 2001)	Optimal reliability design fundamental and application.	١٥
(Sadan, et al., 2003)	Efficiently solving the redundancy allocation problem using tabu search.	١٦
(Zhang, et al., 2005)	Availability and reliability of k-out-of-(MCN):G warm standby systems.	١٧
(Gupta, et al., 2005)	Numerical analysis of reliability and availability of the series processes in butter oil processing plant.	١٨
(Li, et al., 2006)	Reliability analysis of a repairable k-out-of-n system with some components being suspended when the system is down.	١٩
(Barron, et al., 2006)	Analysis of r out of n systems with several repairmen, exponential life times and phase type repair times.	٢٠
(Frostig, et al., 2006)	On the availability of an r out of n repairable systems.	٢١
(Gen, et al., 2006)	Soft computing approach for reliability optimization.	٢٢
(Chen., 2006)	IAs based approach for reliability redundancy allocation problems.	٢٣
(Karin, et al., 2007)	Availability of k-out-of-N systems under block replacement sharing limited spares and repair capacity.	٢٤
(Sharma, et al., 2008)	Availability optimization of a series parallel systems using genetic algorithms.	٢٥

(Juang, et al., 2008)	A knowledge management system for series-parallel availability optimization and design.	۲۶
(Linmin, et al., 2012)	Availability analysis and design optimization for repairable series-parallel system with failure dependencies.	۲۷
(Wenqing, et al., 2014)	Reliability analysis of k-out-of-n:G repairable system with single vacation.	۲۸
(Yuchang, et al., 2015)	Efficient analysis of multi-state k-out-of-n systems.	۲۹

۲-۷. جمع بندی

در فصل دوم به ادبیات موضوع پرداخته شده است یعنی بطور کامل مفاهیم شاخص‌های قابلیت اطمینان ، مدل سازی شبکه‌ها، ارزیابی قابلیت اطمینان انواع سیستم‌ها، مفاهیم تئوری صف، زنجیره مارکف و الگوریتم رقابت استعماری مورد بررسی قرار گرفته است و با توجه به مفاهیم ذکر شده یکی از راههای افزایش دسترس پذیری سیستم‌های تعمیر پذیر استفاده از سیستم‌های k از n و چیدمان کل سیستم بصورت سری می‌باشد. و مدلی که در این پایان نامه مورد بررسی قرار گرفته شده است از نوع سیستم تعمیر پذیر است یعنی پس از هر خرابی تعمیر می‌شود در نیجه از فرآیند تولد و مرگ و همچنین زنجیری مارکف که از مفاهیم پایه‌ای در تئوری صف می‌باشد برای بدست آوردن دسترس پذیری استفاده شده است. و از آنجا که این نوع مسائل در دسته مسائل Np-hard قرار می‌گیرد از الگوریتم رقابت استعماری که الگوریتمی بومی و تقریبا جدیدی است برای حل استفاده شده است.

فصـــــل ســوم

روش شنـــــاسی تحـــقیـــق

۳-۱. مقدمه

پـژوهش یـا تحقیـق، یـک رونـد هوشـمندانه سـازمان یافتـه بـرای یافـت، بـازگیری و بـازنگری پدیـده هـا، رخدادها،رفتارها و انگاشته‌ها است. پژوهش همچنین برای استفاده از پدیده‌های موجود برای دسـت یـافتن بـه راه کارهای عملی و فناوری‌ها بکار می‌رود. پژوهش در دوبعد یافت پرسش پژوهش و پاسخ به آن می‌باشد.

در این فصل ابتدا به مفاهیم روش تحقیق پرداخته و سپس با استفاده مفاهیم بیان شده در فصل دوم به ارائه مدل مسأله و مثال‌های عددی می‌پردازیم.

۳-۲. نوع تحقیق

براساس هدف پژوهش را می‌توان به سه گروه بنیادی، کاربردی و عملی تقسیم کرد:

۱. تحقیق بنیادی: پژوهشی است که به کشف ماهیت اشیاء، پدیده‌ها و روابط بین متغیرها، اصول، قوانین و ساخت یا آزمایش تئوری‌ها و نظریه‌ها می‌پردازد و به توسعه مرزهای دانش کمک می‌کند.
۲. تحقیق کاربردی: پژوهشی است که با استفاده از نتایج تحقیقات بنیـادی بـه منظـور بهبـود و بـه کمـال رساندن رفتارها، روش‌ها، ابزاررها، وسایل، تولیدات، ساختارها و الگوهای مورد استفاده جوامع انسانی انجام می‌شود.
۳. تحقیق عملی: پژوهشی است که با استفاده از نتایج تحقیقات بنیادی و با هدف رفع مسائل و مشـکلات جوامع انسانی انجام می‌شود.

تحقیق حاضر یک مسأله بهینه سازی دسترس پذیری برپایه یک مدل ریاضی است کـه در گـروه تحقیقـات کاربردی قرار می‌گیرد.

۳-۳. روش تحقیق

طرح حاضر از نظر ماهیت روش، تحلیلی توصیفی از جنبه‌ی پیمایشی می‌باشد، بخاطر این که به توصیف نظـام مند ویژگی‌های سیستم‌های k از n پرداخته و پس از مدل سازی دسترس پذیری، بـا اسـتفاده از الگـوریتم رقابـت استعماری به حل مدل می‌پردازد.

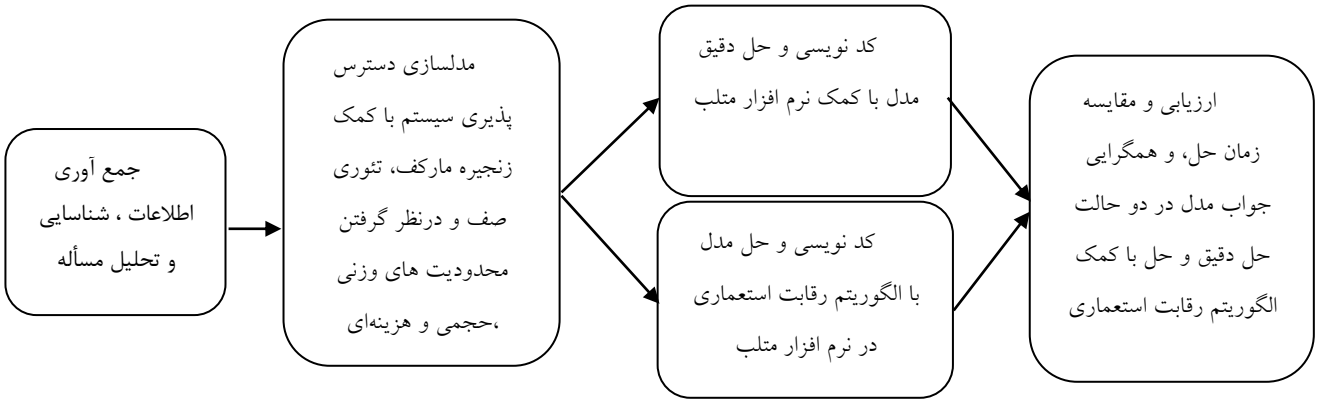
۳-٤. روش جمع‌آوری اطلاعات

روش‌های گردآوری اطلاعات به طور کلی به دو دسته کتابخانه‌ای و میدانی تقسیم می‌شوند. این روش‌ها در دو فضای واقعی و مجازی قابل انجام و کاربرد می‌باشند به عبارتی اطلاعات و داده‌های مورد نیاز پژوهش را می‌توان هم از فضای واقعی و هم از فضای مجازی و یا هر دو فضا با یک یا چند روش مناسب گردآوری و تهیه کرد. روش‌های فضای واقعی همان روش‌های سنتی است که محقق با استفاده از روش‌های کتابخانه‌ای و یا میدانی به جمع‌آوری اطلاعات می‌پردازد. روش فضای مجازی روش جدیدی است که با توسعه شبکه‌های اینترنت و فضای مجازی در اختیار پژوهشگران قرار دارد و آنها می‌توانند روش‌های کتابخانه‌ای و میدانی سنتی را برای گردآوری اطلاعات در فضای مجازی و اینترنت مورد استفاده قرار دهند (حافظ‌نیا ۱۳۸۹).

برای تحقیق حاضر از روش کتابخانه‌ای در فضای واقعی و مجازی برای جمع‌آوری اطلاعات استفاده شده است.

۳-۵. روش تجزیه و تحلیل اطلاعات

روش تجزیه و تحلیل اطلاعات در شکل زیر نشان داده شده است.



شکل ۳-۱(مدل تجزیه و تحلیل اطلاعات)

۳-٦. مدل ریاضی سیستم k از n تعمیرپذیر

در این فصل ما در ابتدا مدل یک سیستم k از n تعمیرپذیر با چند تعمیر کار را شرح می‌دهیم. چنین مدلی ما را قادر می‌سازد اندازه‌های عملکردی سیستم همچون میاتگین زمان تا خرابی، دستسرس پذیری پایا و میانگین زمان بین خرابی را بررسی کنیم.

۳-٦-۱ . فرضیات مدل

- سیستم دارای ساختار k از n با اجزاء فعال است.
- تمامی اجزاء دارای طول عمر نمایی، مستقل و همسان هستند.
- r تعمیر کار همسان در دسترس وجود دارد ($1 \leq r \leq n-k+1$). تنها تعمیر کار می‌بایست به یک جزء خراب تخصیص داده شود و توزیع زمان تعمیر نمایی، مستقل و همسان هستند.
- هر زمان یک جزء خرا شود، اگر تعمیر کار در دسترس باشد بدون فوت زمان شروع به کار می‌کند و اجزاء براساس این که هر کدام زودتر خراب شدند زودتر سرویس می‌گیرند.
- سیستم زمانی از کار می‌افتد که تعداد اجزاء خراب به n-k+1 برسد.
- در حالیکه سیستم از کار افتاده است، اجزاء بیشتری خراب نمی‌شوند.
- حالت سیستم بر مبنای تعداد اجزاء خراب در سیستم که چه منتظر دریافت سرویس تعمیر هستند و چه آنهایی که در حال تعمیر هستند تعریف می_شوند.
- احتمال این که دو جز یا بیشتر بطور همزمان و یا در یک فاصله زمانی خیلی کوتاه خراب شوند و یا به حالت برگشت درآیند قابل چشم پوشی و نزدیک به صفر است.

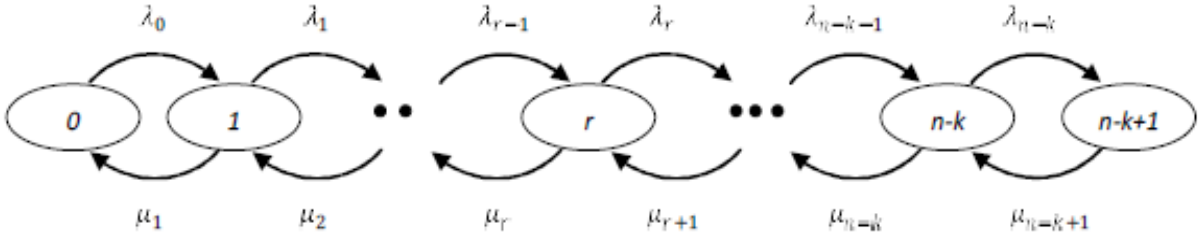
۳-٦-۲ . نمادها

- i :تعداد اجزاء خراب در سیستم
- t :زمان
- λ_i : نرخ خرابی سسیستم زمانی‌که به تعداد i جزء خراب در سیستم وجود داشته باشد ($0 \leq i \leq n-k$).

- μ_i : نـرخ خرابـی سسیسـتم زمـانی‌کـه بـه تعـداد i جـزء خـراب در سیسـتم وجـود داشـته باشـد $(1 \le i \le n-k+1)$.
- $P_i(t)$: احتمال این که i جزء خراب در زمان t در سیستم وجود داشته باشد $(0 \le i \le n-k+1)$.
- π_i : احتمال این که در دراز مدت i جزء خراب در سیستم وجود داشته باشد(احتمال حالت پایا).

۳-۶-۳ . معادلات تعادلی و دسترس پذیری سیستم k از n تعمیر پذیر

با استفاده از مـدل مـارکف و توضـیحات داده شـده در بـالا، دیـاگرام انتقـال حالـت سیسـتم بصـورت شـکل ۳-۲(دیاگرام انتقال یک سیستم با ساختار k از n) خواهد بود.



شکل ۳-۲(دیاگرام انتقال یک سیستم با ساختار k از n)

اعداد داخل دایره‌ها نشان دهنده حالت سیستم است. برای مثال حالت *n-k+1* بیـانگر ایـن موضـوع اسـت کـه *n-k+1* جزء خراب در سیستم وجود دارد و در این حالت طبق تعریـف سیسـتم‌هـای k از n ایـن سیسـتم از کـار خواهد افتاد. همانطور که می‌دانیم در سیستم k از n تعمیر پذیر تعداد اجزاء خراب در سیستم تغییر می‌کند و وقتـی سیستم از کار می‌افتد تمامی تعمیرکاران برای تعمیر اجزاء خراب مورد استفاده قرار می‌گیرند، و بمحـض ایـن کـه تعداد اجزاء خراب سیستم به کمتر از *n-k+1* می‌رسد سیستم دوباره شروع بکار می‌کند. بنابراین حالت سیسـتم در طی زمان بین حالت خراب و سالم تغییر می‌کند. و احتمال این که سیستم در زمان t در حال کار کردن باشـد(تعداد اجزاء خراب سیستم در زمان t کمتر از *n-k+1* باشد) را دسترس پذیری سیستم می‌گویند و به بیان ریاضی دسترس پذیری عبارت است از (احتمال این که در زمان t ، *n-k+1* جزء خراب در سیستم باشد – ۱):

$$A_s = 1 - p_{n-k+1}(t) \qquad (۱-۳)$$

همانطور که در تعاریف تئوری صف آورده شده است، π_i معرف بودن i مشتری (جزء خـراب) در سیسـتم یـا درصدی از زمان سیستم است که سیستم دارای i مشتری (جزء خراب)است (در دراز مدت)، یا

$$\pi_i = \lim_{t\to\infty} p[N(t) = i] \qquad i = 0,1,\dots,n-k+1 \qquad (۲-۳)$$

برای محاسبه π_i در فرآیند تولد و مرگ، شکل ۳-۲(دیاگرام انتقال یک سیستم با ساختار k از n بکار گرفته می-شود. ابتدا از حالت صفر که هیچ مشتری (جزء خراب) در سیستم وجود ندارد، شروع می‌کنیم. همانطور که نمودار آهنگ نشان می‌دهد حالت سیستم فقط در صورتی تغییر می‌کند که یک مشتری جدید وارد شود(یک خرابی رخ دهد). در این‌صورت، حالت سیستم نیز"۱" خواهد شد. تغییر حالت سیستم از صفر به ۱ یعنی ورود یک مشتری(رویداد یک خرابی) با آهنگ λ_0 انجام شود. به همین ترتیب، با در نظر گرفتن این مطلب که ورود به حالت صفر نیز فقط با تغییر حالت سیستم از یک به صفر امکان پذیر است، آهنگ ورود به حالت نیز برابر μ_1 خواهد بود. در نتیجه، معادله تعادلی در حالت صفر برابر است با:

$$\lambda_0\pi_0 = \mu_1\pi_1 \qquad (۳-۳)$$

اکنون حالت ۱ را در نظر بگیرید. تغییر حالت سیستم، به دو طریق، ورود مشتری جدید(خراب شدن یک جزء) با آهنگ $\lambda_۱$ یا خروج تنها مشتری داخل سیستم(تعمیر جزء خراب) با آهنگ μ_1 میسر است. از طرف دیگر، از دو طریق نیز می‌توان به حالت ۱ رسید. از حالت صفر با ورود یک مشتری(خراب شدن یک جزء) یا از حالت ۲ با خروج یکی از مشتری‌ها از داخل سیستم(تعمیر جزء خراب). بنابراین معادله تعادلی برای این حالت عبارت است از:

$$(\lambda_1 + \mu_1)\pi_1 = \lambda_0\pi_0 + \mu_2\pi_2 \qquad (٤-۳)$$

به همین ترتیب، معادله تعادلی برای حالات $1 \le i \le n-k$ *به شرح زیر خواهد بود:*

$$(\lambda_i + \mu_i)\pi_i = \lambda_{i-1}\pi_{i-1} + \mu_{i+1}\pi_{i+1} \qquad (۵-۳)$$

با استفاده از رابطه‌های (۳-۳)، (۳-٤) و (۳-۵)نتیجه می‌شود:

$$\pi_1 = \frac{\lambda_0}{\mu_1}\pi_0 \qquad (٦-۳)$$

$$\pi_2 = \frac{\lambda_0\lambda_1}{\mu_1\mu_2}\pi_0 \qquad (۷-۳)$$

.

.

.

$$\pi_i = \frac{\lambda_0\lambda_1\cdots\lambda_{i-1}}{\mu_1\mu_2\cdots\mu_i}\pi_0, \qquad for\ i = 1,2,3,\dots,n-k+1 \qquad (۸-۳)$$

از طرف دیگر می‌دانیم که $\sum_{i=0}^{i=n-k+1} \pi_i = 1$، بنابراین خواهیم داشت:

$$\pi_0 = \left(1 + \sum_{i=1}^{n-k+1} \frac{\lambda_0 \lambda_1 \cdots \lambda_{i-1}}{\mu_1 \mu_2 \cdots \mu_i}\right)^{-1} \quad (۹-۳)$$

در نتیجه برای π_i خواهیم داشت:

$$\pi_i = \frac{\lambda_0 \lambda_1 \cdots \lambda_{i-1}}{\mu_1 \mu_2 \cdots \mu_i} \left(1 + \sum_{i=1}^{n-k+1} \frac{\lambda_0 \lambda_1 \cdots \lambda_{i-1}}{\mu_1 \mu_2 \cdots \mu_i}\right)^{-1} \; for\; i = 1,2,3, \dots, n-k+1 \quad (۱۰-۳)$$

با توجه به رابطه(۳-۱۰)احتمال این که در بلند مدت $n\text{-}k+1$ خرابی در سیستم وجود داشته باشد(یعنی سیستم همیشه خراب باشد) برابر است با:

$$\pi_{n-k+1} = \frac{\lambda_0 \lambda_1 \cdots \lambda_{n-k}}{\mu_1 \mu_2 \cdots \mu_{n-k+1}} \left(1 + \sum_{i=1}^{n-k+1} \frac{\lambda_0 \lambda_1 \cdots \lambda_{i-1}}{\mu_1 \mu_2 \cdots \mu_i}\right)^{-1} \quad (۱۱-۳)$$

نهایتا با استفاده از رابطه‌های (۳-۱) و (۳-۱۱)، دسترس پذیری پایای سیستم بدست خواهد آمد:

$$A_s(t) = 1 - p_{n-k+1}(t) = 1 - \left[\frac{\lambda_0 \lambda_1 \cdots \lambda_{n-k}}{\mu_1 \mu_2 \cdots \mu_{n-k+1}} \left(1 + \sum_{i=1}^{n-k+1} \frac{\lambda_0 \lambda_1 \cdots \lambda_{i-1}}{\mu_1 \mu_2 \cdots \mu_i}\right)^{-1}\right] \quad (۱۲-۳)$$

۳-٦-٤ . مدل مسأله

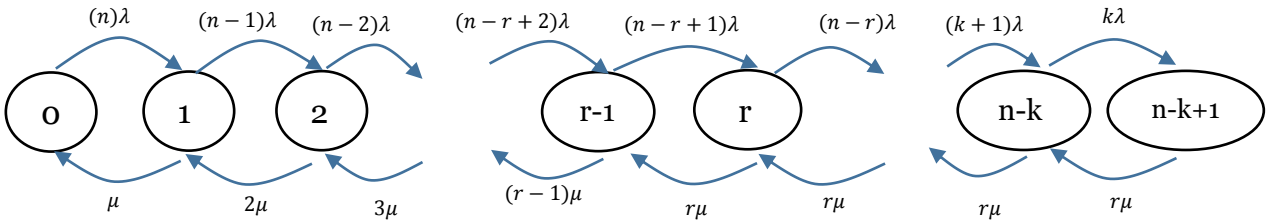
همانطور که می‌دانیم تعداد مشتریان بالقوه سیستم متناهی است و حداکثر تعداد مشتری‌هایی که به سیستم مراجعه می‌کنند برابر با عدد متناهی، مثلا n، است. یکی از کاربردهای این مدل در برنامه ریزی نگهداری و تعمیرات است. اگر تعداد تعمیرکاران را r و تعداد اجزای که ممکن است خراب شود را n فرض کنیم، با یک سیستم صف سر و کار داریم که می‌توان در آن اجزاء را مشتری و تعمیرکاران را خدمت دهنده فرض کرد. بدیهی است که در این سیستم جمعیت مشتریان محدود و برابر با n است. این سیستم را در صورتی می‌توان یک مدل نمایی با جمعیت متناهی نامید که هم مدت زمان تعمیر اجزاء و هم مدت زمانی که طول می‌کشد تا یک جزء خراب شود را نمایی فرض کنیم. در هر لحظه اجزاء را می‌توان به دو گروه تقسیم کرد. گروه اول گروه اجزایی که خراب شده‌اند و تحت تعمیر و یا منتظر تعمیر هستند. گروه دوم اجزایی که سالم هستند ولی در هر لحظه ممکن است خراب شوند. مدت زمان سالم بودن هر جزء قبل از خراب شدن را متغیر تصادفی با توزیع نمایی و پارامتر λ فرض می‌کنیم. اگر تعداد اجزاء خراب برابر i باشد، مجموعا (n-i) جزء سالم هستند که مشتری‌های بعدی سیستم محسوب می‌شوند. دراین حالت زمان ورود اولین مشتری، در واقع زمانی است که اولین جزء سالم خراب می‌شود. بنابراین مدت زمانی که طول می‌کشد تا اولین جزء خراب شود یک متغیر تصادفی و برابر با حداقل (n-i) متغیر

تصادفی نمایی است. همانطور که می‌دانیم چنین متغیر تصادفی دارای توزیع نمایی با پارامتر $(n-i)\lambda$ است(مدرس و همکاران، ۱۳۹۱). بنابراین خواهیم داشت:

$$\lambda_i = (n-i)\lambda \qquad 0 \le i \le n-k \tag{۱۳-۳}$$

$$\mu_i = \begin{cases} i\mu & 0 \le i \le r \\ r\mu & r \le i \le n-k+1 \end{cases} \tag{۱٤-۳}$$

با توجه به نکات گفته شده، نمودار آهنگ مساله بصورت شکل ۳-۳ (نمودار آهنگ انتقال مدل مسأله) درمی‌آید:



شکل ۳-۳ (نمودار آهنگ انتقال مدل مسأله)

با توجه به شکل ۳-۳ (نمودار آهنگ انتقال مدل مسأله) معادلات تعادلی برای حالات مختلف(حالت صفر تا n-k+1)بصورت زیر خواهد شد.

حالت صفر:

$$\pi_1 = n(\frac{\lambda}{\mu})\,\pi_0 \tag{۱۵-۳}$$

حالت ۱:

$$\pi_2 = \frac{n(n-1)}{2}(\frac{\lambda}{\mu})^2\,\pi_0 \tag{۱٦-۳}$$

.

.

.

به ازای $i \le r$:

$$\pi_i = \frac{n!}{(n-i)!i!}(\frac{\lambda}{\mu})^i\,\pi_0 \tag{۱۷-۳}$$

.

.

.

حالت r :

$$\pi_{r+1} = \frac{n!}{(n-r-1)!r!r}\left(\frac{\lambda}{\mu}\right)^{r+1}\pi_0 \qquad (۱۸-۳)$$

.

.

.

و به ازای $r+1 \le i \le n-k+1$:

$$\pi_i = \frac{n!}{(n-i)!r!r^{i-r}}\left(\frac{\lambda}{\mu}\right)^{i}\pi_0 \qquad (۱۹-۳)$$

از طرف دیگر می‌دانیم که $\sum_{i=0}^{i=n-k+1}\pi_i = 1$، بنابراین خواهیم داشت:

$$\pi_0 + \pi_1 + \pi_2 + \cdots + \pi_r + \pi_{r+1} + \cdots + \pi_{n-k+1} = 1 \qquad (۲۰-۳)$$

و با جایگذاری روابط (۳-۱۵)، (۳-۱۶)، (۳-۱۷)، (۳-۱۸) و (۳-۱۹) در رابطه (۳-۲۰) خواهیم داشت:

$$\pi_0 + \left(\sum_{i=1}^{r}\frac{n!}{(n-i)!\,i!}\left(\frac{\lambda}{\mu}\right)^{i}\pi_0 + \sum_{i=r+1}^{n-k+1}\frac{n!}{(n-i)!\,r!\,r^{i-r}}\left(\frac{\lambda}{\mu}\right)^{i}\pi_0\right) = 1$$
$$= \pi_0\left(1 + \sum_{i=1}^{r}\frac{n!}{(n-i)!\,i!}\left(\frac{\lambda}{\mu}\right)^{i} + \sum_{i=r+1}^{n-k+1}\frac{n!}{(n-i)!\,r!\,r^{i-r}}\left(\frac{\lambda}{\mu}\right)^{i}\right) \qquad (۲۱-۳)$$

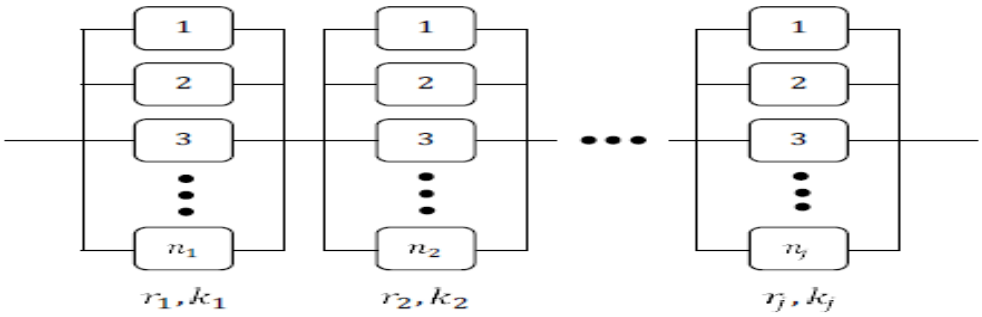
و π_0 بصورت رابطه زیر در خواهد آمد:

$$\pi_0 = \left(1 + \sum_{i=1}^{r}\frac{n!}{(n-i)!\,i!}\left(\frac{\lambda}{\mu}\right)^{i} + \sum_{i=r+1}^{n-k+1}\frac{n!}{(n-i)!\,r!\,r^{i-r}}\left(\frac{\lambda}{\mu}\right)^{i}\right)^{-1} \qquad (۲۲-۳)$$

و سرانجام با استفاده از رابطه (۳-۱۹) و (۳-۲۲)، π_{n-k+1} بصورت زیر محاسبه می‌شود:(۳-۲۰)

$$\pi_{n-k+1} = \left(1 + \sum_{i=1}^{r}\binom{n}{i}\left(\frac{\lambda}{\mu}\right)^{i} + \sum_{i=r+1}^{n-k+1}\frac{n!}{(n-i)!\,r!\,r^{i-r}}\left(\frac{\lambda}{\mu}\right)^{i}\right)^{-1}\frac{n!}{(k-1)!\,r!\,r^{n-k-r+1}}\left(\frac{\lambda}{\mu}\right)^{n-k+1} \qquad (۲۳-۳)$$

سیستم مورد نظر ما با توجه به شکل ۳-٤(ساختار سیستم مسأله) یک سیستم سری است که دارای زیر سیستم-هایی(j=1,2,….,y) است که هر کدام از این زیر سیستم‌ها از ساختار k از n تعمیر پذیر پیروی می‌کند،



شکل ۳-۴(ساختار سیستم مسأله)

و دسترس پذیری سیستم بوسیله رابطه زیر تعیین می‌شود(Juang, et al., 2008):

$$A_s = \prod_{j=1}^{y} Aj \qquad (۲٤-۳)$$

با استفاده از رابطه (۳-۱) و (۳-۲۳) دسترس پذیری برای یک زیر سیستم بصورت زیر بدست خواهد آمد:

$$A_s = \left[1 - \left(1 + \sum_{i=1}^{r} \binom{n}{i} \left(\frac{\lambda}{\mu}\right)^i + \sum_{i=r+1}^{n-k+1} \frac{n!}{(n-i)!\, r!\, r^{i-r}} \left(\frac{\lambda}{\mu}\right)^i\right)^{-1} \frac{n!}{(k-1)!\, r!\, r^{n-k-r+1}} \left(\frac{\lambda}{\mu}\right)^{n-k+1}\right] \qquad (۲۵-۳)$$

حال با استفاده از رابطه (۳-۲٤) و (۳-۲۵) دسترس پذیری کل سیستم برابر خواهد بود با:

$$\boldsymbol{A_s = \prod_{j=1}^{y} \left[1 - \left(1 + \sum_{i=1}^{r_j} \binom{n_j}{i} \left(\frac{\lambda}{\mu}\right)^i + \sum_{i=r_j+1}^{n_j-k_j+1} \frac{n_j!}{(n_j-i)!\, r_j!\, r_j^{i-r_j}} \left(\frac{\lambda}{\mu}\right)^i\right)^{-1} \frac{n_j!}{(k_j-1)!\, r_j!\, r_j^{n_j-k_j-r_j+1}} \left(\frac{\lambda}{\mu}\right)^{n_j-k_j+1}\right]} \qquad (۲٦-۳)$$

هدف این مسأله بهینه سازی دسترس پذیری سیستم در حالت پایا با توجه به محدودیت‌های وزنی، هزینه‌ای، حجمی است. همچنین متغیرهای تصمیم مسأله تعداد تعمیرکاران و تعداد اجزاء افزونه برای هر زیر سیستم است بطوریکه محدودیت‌های مسأله را ارضا کرده و تابع هدف مسأله که دسترس پذیری سیستم است را ماکزیمم کند.

با توجه به رابطه (۳-۲٦)، مدل ریاضی مسأله با در نظر گرفتن محدودیت‌ها به صورت شکل زیر خواهد بود:

$$Max\ A_s = \prod_{j=1}^{y}\left[1 - \left(1 + \sum_{i=1}^{r_j}\binom{n_j}{i}\left(\frac{\lambda}{\mu}\right)^i + \sum_{i=r_j+1}^{n_j-k_j+1}\frac{n_j!}{(n_j-i)!\,r_j!\,r_j{}^{i-r_j}}\left(\frac{\lambda}{\mu}\right)^i\right)^{-1}\frac{n_j!}{(k_j-1)!\,r_j!\,r_j{}^{n_j-k_j-r_j+1}}\left(\frac{\lambda}{\mu}\right)^{n_j-k_j+1}\right]$$

Subject to:

$$\sum_{j=1}^{y}(c_j n_j + r_j h_j) \le c \qquad (۲۷-۳)$$

$$\sum_{j=1}^{y} w_j n_j \le w \qquad (۲۸-۳)$$

$$\sum_{j=1}^{y} v_j n_j \le v \qquad (۲۹-۳)$$

$$\forall j,\ j = 1,2,\dots,y \qquad 1 \le r_j \le (n_j - k_j + 1) \qquad (۳۰-۳)$$

$$\forall \boldsymbol{j},\ \boldsymbol{j = 1,2,\dots,y} \qquad \boldsymbol{n_j \ge k_j} \qquad (۳۱-۳)$$

شکل ۳-۵(مدل ریاضی مسأله)

مدل ریاضی مسأله شامل تابع هدف که حداکثر دسترس پذیری سیستم است. این مدل شامل پنج محدودیت است. محدودیت (۳-۲۷) در واقع محدودیت بودجه‌ای مسأله است، c بودجه کل سیستم است و c_j هزینه یک در زیر سیستم j است و h_j هم هزینه به خدمت گیری تعمیرکاران برای زیر سیستم j است. محدودیت (۳-۲۸) محدودیت وزنی سیستم است، w حد بالای وزنی سیستم است. و w_j وزن هر جز در زیر سیستم j است. محدودیت (۳-۲۹) محدودیت حجمی مسأله است که حد بالای حجم سیستم v است. و v_j وزن هر جز در زیر سیستم j است. محدودیت (۳-۳۰) حد بالا و پایین تعداد تعمیرکاران برای خدمت‌گیری را نشان می‌دهد و محدودیت (۳-۳۱) در واقع منطق سیستم k از n را نشان می‌دهد که همیشه می‌بایست برقرار باشد تا یک ساختار k از n تعریف شود.

آنچه از مطالب بالا مشخص است این است که مسأله ما یک مسأله تخصیص افزونگی است که اجزاء تعمیر پذیر هستند و با توجه به محدویت هزینه‌ای سیستم تعداد تعمیرکاران به خدمت گرفته شده در مدت زمان مود نظر جزو متغیرهای تصمیم مسأله است.

همانطور که از مدل ریاضی مسأله پیدا است این مساله با توجه به این که تابع هدف یک تابع غیر خطی است و فضای جواب و متغیر تصمیم مسأله عدد صحیح است، این مسأله از نوع برنامه‌ریزی غیر خطی عدد صحیح است.

۳-٦-٥ . مدل‌های عددی مسأله

در بخش ۳-٦-٤ مدل ریاضی مسأله مورد نظر را بیان کردیم، مسأله مورد نظر ما شامل یک سیستم سری با زیر سیستم‌هایی که ساختار آن k از n است می‌باشد. هدف مسأله پیدا کردن تعداد افزونه‌ها و تعمیرکاران مورد نیاز هر زیر سیستم با توجه به محدودیت‌های حجمی، وزنی و هزینه‌ای بمنظور حداکثر دسترس‌پذیری است. برای روشن تر شدن موضوع به ارائه دو مثال عددی می‌پردازیم و در فصل بعد به نتایج حل آنها که با نرم افزار MATLAB R2013a صورت گرفته و در پیوست آورده شده است می‌پردازیم.

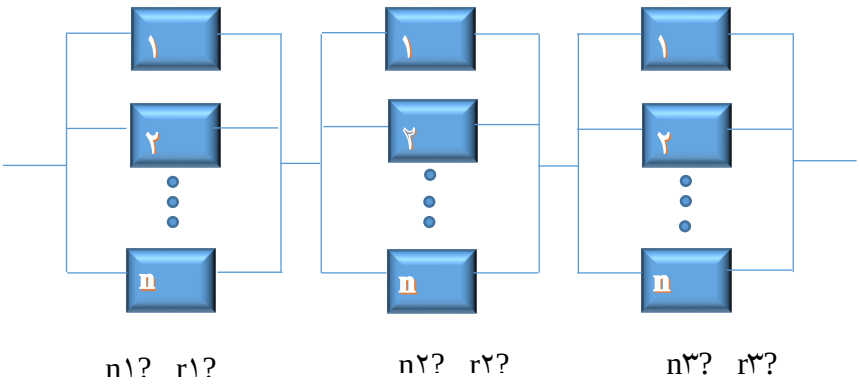
۳-٦-٥-١ . مثال ۱:

مثال ۱ شامل یک سیستم با سه زیر سیستم است که پارامترهای ورودی در جدول ۳-۱(پارامترهای ورودی مثال ۱) نشان داده شده است:

جدول ۳-۱(پارامترهای ورودی مثال ۱)

j	w_j	v_j	c_j	h_j	k_j
۱	۰.۳	۱.۱	۵	۱	۲
۲	۰.٤	۱.۳	٤	۱	۱
۳	۰.۲	۲.۱	۷	۲	۲
w	V	C	Y	μ	λ
٤٥	٤۰	٤۸۰	۳	۱.۱	۱.٤

همانطور که در شکل ۳-٦ (ساختار سیستم در مثال ۱) نشان داده شده است این مسأله یک سیستم را شرح می دهد که سه زیر سیستم دارد ($Y=3$) و بودجه کل سیستم $C=480$ واحد پولی است و حد مجاز وزنی برای سیستم $W=45$ واحد وزنی ، و حد مجاز حجمی سیستم $V=40$ واحد حجمی می‌باشد و حداقل جزء برای زیر سیستم‌ها برای اینکه هر زیر سیستم کار کند برای زیر سیستم اول ۲، برای زیر سیستم دوم ۱، و برای زیر سیستم سوم ۲ می باشد($k=[2\ 1\ 2]$).



شکل ۳-۶ (ساختار سیستم در مثال ۱)

ضرایب وزنی، حجمی، هزینه‌ای و هزینه به ازای یک تعمیرکار برای هر زیر سیستم به ترتیب برابر با w, v, c, h می‌باشد که مقادیر آن در جدول ۳-۱(پارامترهای ورودی مثال ۱) نشان داده شده است. بنابراین با توجه به روابط (۳-۲۷)، (۳-۲۸)، (۳-۲۹)، (۳-۳۰) و (۳-۳۱) محدودیت‌های مدل مثال ۱ بصورت زیر خواهد بود:

$$5n_1 + 4n_2 + 7n_3 + r_1 + r_2 + 2r_3 \leq 480 \qquad (۳۲-۳)$$

$$0.3n_1 + 0.4n_2 + 0.2n_3 \leq 45 \qquad (۳۳-۳)$$

$$1.1n_1 + 1.3n_2 + 2.1n_3 \leq 40 \quad (۳٤-۳)$$

$$1 \leq r_1 \leq n_1 - 1 \quad (۳٥-۳)$$

$$1 \leq r_2 \leq n_2 \quad (۳٦-۳)$$

$$1 \leq r_3 \leq n_3 - 1 \quad (۳۷-۳)$$

$$\boldsymbol{n_1 \geq 2} \quad (۳۸-۳)$$

$$\boldsymbol{n_2 \geq 1} \quad (۳۹-۳)$$

$$\boldsymbol{n_3 \geq 2} \quad (٤۰-۳)$$

۳-٦-٥-۲ . مثال ۲:

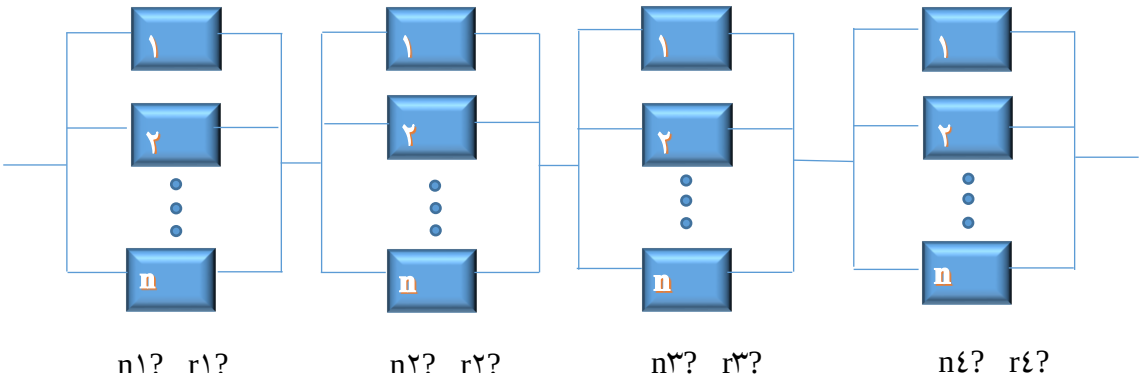
مثال ۲ نیز شامل یک سیستم با چهار زیر سیستم است که پارامترهای ورودی آن در جدول ۳-۲ (پارامترهای ورودی مثال ۲) نشان داده شده است:

جدول ۳-۲ (پارامترهای ورودی مثال ۲)

j	w_j	v_j	c_j	h_j	k_j
۱	۰.۳	۱.۱	٥	۱	۲
۲	۰.٤	۱.۳	٤	۱	۱
۳	۰.۲۷	۲.۱	۷	۲	۲
٤	۰.۷	۳.۱	٦	۲	۳
w	V	C	Y	μ	λ
۳٥	٥٥	۲٦۱	٤	۱.۱	۱.٤

همانطور که در شکل ۳-۷(ساختار سیستم در مثال ۲) نشان داده شده است این مسأله یک سیستم را شرح می دهد که چهار زیر سیستم دارد (Y=4) و بودجه کل سیستم C=261 واحد پولی است و حد مجاز وزنی برای سیستم W=35 واحد وزنی ، و حد مجاز حجمی سیستم V=55 واحد حجمی می‌باشد و حداقل جزء برای زیر

سیستم¬ها برای اینکه هر زیر سیستم کار کند برای زیر سیستم اول ۲، برای زیر سیستم دوم ۱، برای زیـر سیسـتم سوم ۲ و برای زیر سیستم چهارم ۳ می باشد(k=[2 1 2 3]).



شکل ۳-۷(ساختار سیستم در مثال ۲)

ضرایب وزنی، حجمی، هزینه‌ای و هزینه به ازای یک تعمیرکار برای هر زیر سیستم به ترتیب برابر با w, v, c, h می‌باشد که مقادیر آن در جدول ۳-۲ (پارامترهای ورودی مثال ۲) نشان داده شده است. بنابراین با توجه بـه روابـط (۳-۲۷)، (۳-۲۸)، (۳-۲۹)، (۳-۳۰) و (۳-۳۱) محدودیت‌های مدل مثال ۲ بصورت زیر خواهد بود:

$$5n_1 + 4n_2 + 7n_3 + 6n_4 + r_1 + r_2 + 2r_3 + 2r_4 \leq 261 \qquad (۳-٤۱)$$

$$0.3n_1 + 0.4n_2 + 0.27n_3 + 0.7n_4 \leq 35 \qquad (۳-٤۲)$$

$$1.1n_1 + 1.3n_2 + 2.1n_3 + 3.1n_4 \leq 55 \qquad (۳-٤۳)$$

$$1 \leq r_1 \leq n_1 - 1 \qquad (۳-٤٤)$$

$$1 \leq r_2 \leq n_2 \qquad (۳-٤٥)$$

$$1 \leq r_3 \leq n_3 - 1 \qquad (۳-٤٦)$$

$$1 \leq r_4 \leq n_4 - 2 \qquad (۳-٤۷)$$

$$n_1 \geq 2 \qquad (۳-٤۸)$$

$$n_2 \geq 1 \qquad (۳-٤۹)$$

$$n_3 \geq 2 \qquad (۳-۵۰)$$

$$n_4 \geq 3 \qquad (۳-۵۱)$$

فصل چهارم

یافته‌های تحقیق

٤-١. مقدمه

در این فصل به حل مسائل ارائه شده در فصل قبل با نرم افزار MATLAB R2013a می‌پردازیم در واقع با استفاده از این نرم‌افزار کد مسأله را نوشته و سپس به دو روش حل خواهیم کرد. روش اول جستجوی تمام نقاط تا بدست آوردن نقطه بهینه و روش دوم استفاده از الگوریتم رقابت استعماری می‌باشد. سپس به تحلیل و مقایسه خواهیم پرداخت.

٤-٢. حل دقیق مسأله

برای حل دقیق مسأله، مثال های عددی ارائه شده در فصل سوم (مثال۱ و مثال۲) را توسط نرم افزار متلب کد کرده‌ایم، که کد این مسائل در پیوست آورده شده است.

این کد بدین صورت نوشته شده است که تمامی نقاط در فضا، یعنی تمامی حالات مختلف برای n و r که فضای مسأله را تشکیل می‌دهند، ابتدا در محدودیت‌های مسأله قرار داده و در صورت ارضا کردن محدودیت‌ها آن را در تابع هدف قرار می‌دهد، و مقدار آن را در یک متغیر ذخیره می‌کند. در پایان بیشترین مقدار که دسترس‌پذیری سیستم می‌باشد به همراه نقطه بهینه به ما می‌دهد.

٤-٢-١ . نتایج حل دقیق مثال ١

نتایج حاصل از حل توسط نرم افزار متلب در جدول زیر آورده شده است.

جدول ٤-١ (نتایج حل دقیق مثال ١ توسط نرم افزار متلب)

Elapsed time	FINAL_PARAMETER	FINAL_AS
4618.128940	*12 6 9 11 6 8*	*0.92347*

همانطور که در جدول ٤-١ (نتایج حل دقیق مثال ١ توسط نرم افزار متلب) نشان داده شده است نقطه بهینه بصورت زیر بدست آمده است:

$$n = [n_1 = 12, \quad n_2 = 6, \quad n_3 = 9\ ]$$

$$r = [r_1 = 11, \quad r_2 = 6, \quad r_3 = 8\ ]$$

بیشنه دسترس پذیری برابر با ۰.۹۲۳٤۷، و همچنین مدت زمان صرف شده در نرم افزار جهت رسیدن به جواب بهینه ۱.۲۸ ساعت می‌باشد.

بطور خلاصه برای سیستم ذکر شده در مثال ۱ اگر برای زیر سیستم اول ۱۲ افزونه و ۱۱ تعمیر کار، برای زیرسیستم دوم ٦ افزونه و ٦ تعمیرکار و برای زیرسیستم سوم ۹ افزونه و ۸ تعمیرکار در نظر بگریم با توجه به محدودیت‌های حجمی، وزنی و هزینه‌ای به بیشترین دسترس‌پذیری که برابر با ۰.۹۲۳٤۷ می‌باشد دست خواهیم یافت.

٤-۲-۲ . نتایج حل دقیق مثال ۲

نتایج حاصل از حل توسط نرم افزار متلب در جدول زیر آورده شده است.

جدول ٤-۲(نتایج حل دقیق مثال ۲ توسط نرم افزار متلب)

Elapsed time	FINAL_PARAMETER	FINAL_AS
521270.446162	*8 5 7 8 7 5 6 6*	*0.6571*

همانطور که در جدول ٤-۲(نتایج حل دقیق مثال ۲ توسط نرم افزار متلب) نشان داده شده است نقطه بهینه بصورت زیر بدست آمده است:

$$n = [n_1 = 8, \quad n_2 = 5, \quad n_3 = 7, \quad n_4 = 8\ ]$$
$$r = [r_1 = 7, \quad r_2 = 5, \quad r_3 = 6, \quad r_4 = 6\ ]$$

بیشنه دسترس پذیری برابر با ۰.٦٥۷۱، و همچنین مدت زمان صرف شده در نرم افزار جهت رسیدن به جواب بهینه ۱٤٤.۸ ساعت می‌باشد.

بطور خلاصه برای سیستم ذکر شده در مثال ۲ اگر برای زیر سیستم اول ۸ افزونه و ۷ تعمیر کار، برای زیرسیستم دوم ٥ افزونه و ٥ تعمیرکار، برای زیرسیستم سوم ۷ افزونه و ٦ تعمیرکار و برای زیرسیستم چهارم ۸ افزونه و ٦ تعمیرکار در نظر بگیریم با توجه به محدودیت‌های حجمی، وزنی و هزینه‌ای به بیشترین دسترس‌پذیری که برابر با ۰.٦٥۷۱ می‌باشد دست خواهیم یافت.

٤-٣. حل توسط الگوریتم فرا ابتکاری

در این بخش به ارائه و بررسی نتایج حاصل از حل مسائل (مثال ۱ و مثال۲) که توسط الگوریتم رقابت استعماری در نرم افزار متلب کد نویسی شده و در پیوست آورده شده است می‌پردازیم.

٤-٣-١ . نتایج حل مثال ۱ توسط الگوریتم رقابت استعماری

نتایج حاصل از حل مثال ۱ توسط الگوریتم رقابت استعماری پیشنهادی در جدول زیر آورده شده است.

جدول ٤-٣ (نتایج حل مثال ۱ توسط الگوریتم رقابت استعماری)

Elapsed time	FINAL_PARAMETER	FINAL_AS
9.312757	*12 6 9 11 6 8*	*0.9235*

با توجه به جدول ٤-٣ (نتایج حل مثال ۱ توسط الگوریتم رقابت استعماری) بهترین نقطه بدست آمده توسط الگوریتم بصورت زیر است:

$$n = [n_1 = 12, \quad n_2 = 6, \quad n_3 = 9\ ]$$
$$r = [r_1 = 11, \quad r_2 = 6, \quad r_3 = 8\ ]$$

بیشنه دسترس پذیری برابر با ۰.۹۲۳۵، و همچنین زمان صرف شده برای بدست آوردن این جواب توسط الگوریتم رقابت استعماری در نرم افزار متلب برابر ۹.۳۱۲۷۵۷ ثانیه که معادل ۰.۰۰۲۵۸ ساعت می‌باشد.

٤-٣-٢ . نتایج حل مثال ۲ توسط الگوریتم رقابت استعماری

نتایج حاصل از حل مثال ۲ توسط الگوریتم رقابت استعماری پیشنهادی در جدول زیر آورده شده است.

جدول ٤-٤(نتایج حل مثال ۲ توسط الگوریتم رقابت استعماری)

Elapsed time	FINAL_PARAMETER	FINAL_AS
13.229387	*8 5 7 8 7 5 6 6*	*0.6571*

با توجه به جدول ٤-٤(نتایج حل مثال ۲ توسط الگوریتم رقابت استعماری) بهترین نقطه بدست آمده توسط الگوریتم بصورت زیر است:

$$n = [n_1 = 8, \quad n_2 = 5, \quad n_3 = 7, \quad n_4 = 8\ ]$$

$$r = [r_1 = 7, \quad r_2 = 5, \quad r_3 = 6, \quad r_4 = 6\ ]$$

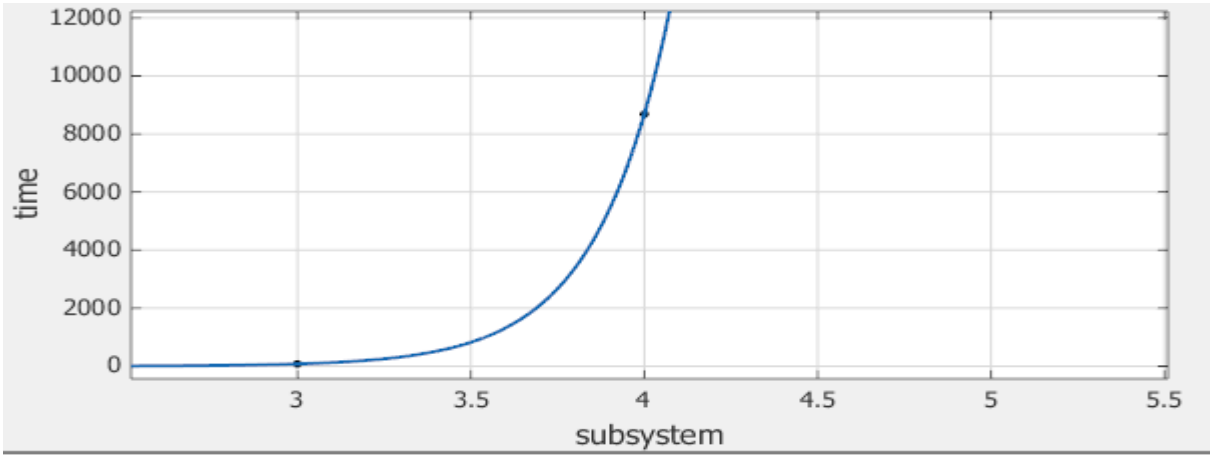
بیشنه دسترس پذیری برابر با ۰.٦٥٧١، و همچنین زمان صرف شده برای بدست آوردن این جواب توسط الگوریتم رقابت استعماری در نرم افزار متلب برابر ۱۳.۲۲۹۳۸۷ ثانیه که معادل ۰.۰۰۳۷ ساعت می‌باشد.

٤-٤. تحلیل و مقایسه

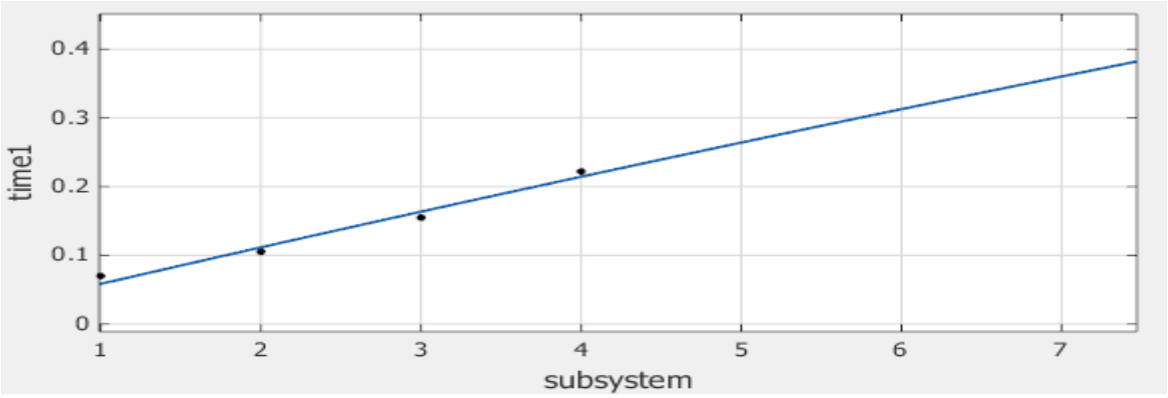
٤-٤-١ . تحلیل زمان صرف شده

با توجه به جدول ٤-١ (نتایج حل دقیق مثال ۱ توسط نرم افزار متلب) و جدول ٤-۲(نتایج حل دقیق مثال ۲ توسط نرم افزار متلب) زمان حل دقیق مسأله در نمودار ٤-۱(زمان صرف شده توسط الگوریتم نشان داده شده است، با افزایش زیر سیستم‌ها فضای جواب مسأله بطور چشمگیری افزایش می‌یابد و با توجه به الگوریتم بیان شده (الگوریتم حل دقیق)، این الگوریتم تمامی نقاط فضا را جستجو می‌کند و بهترین جواب را به ما خواهد داد. و نمودار ٤-۲ (زمان صرف شده توسط الگوریتم رقابت استعماری) زمان صرف شده توسط الگوریتم رقابت استعماری را نشان می‌دهد.

در نمودار ٤-۱(زمان صرف شده توسط الگوریتم و ٤-۲ محور افقی نشان دهنده تعداد زیر سیستم‌ها و محور عمودی نشان دهنده زمان صرف شده به دقیقه است.



نمودار ٤-۱(زمان صرف شده توسط الگوریتم حل دقیق)

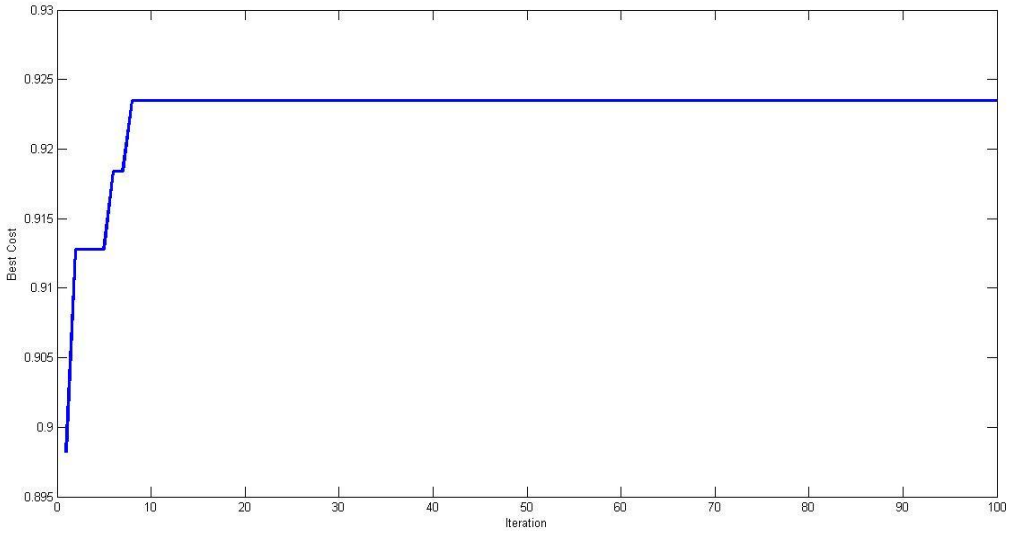


نمودار ٤-٢ (زمان صرف شده توسط الگوریتم رقابت استعماری)

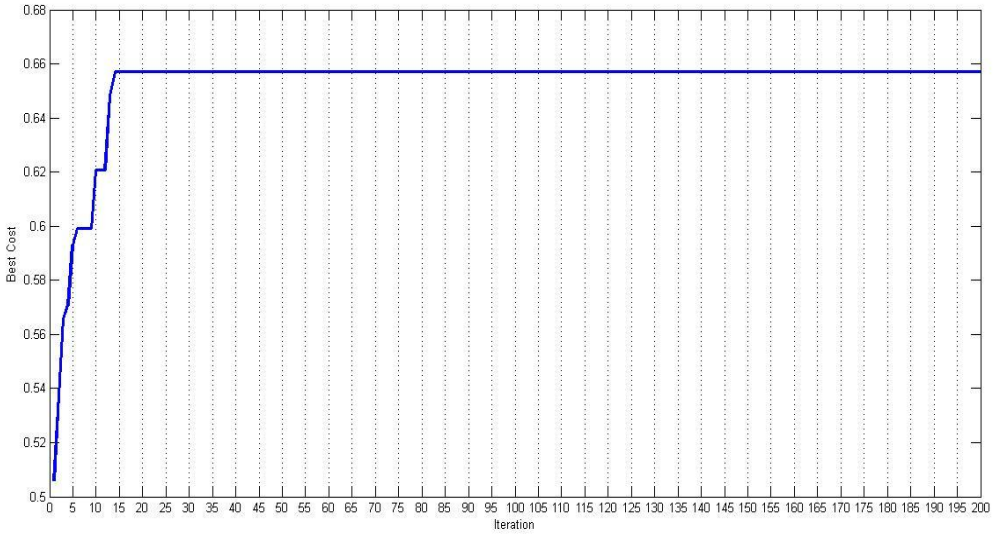
با توجه به این نمودارها با افزایش تعداد زیر سیستم‌ها و افزایش فضای جواب، حل دقیق این مسأله توسط الگوریتم پیشنهادی منطقی بنظر نمی‌رسد.

٤-٤-٢ . تحلیل کارایی و همگرایی الگوریتم رقابت استعماری پیشنهادی

همگرایی الگوریتم پیشنهادی در نمودارهای زیر با توجه به نتایج جدول ٤-٣ (نتایج حل مثال ١ توسط الگوریتم رقابت استعماری) و جدول ٤-٤(نتایج حل مثال ٢ توسط الگوریتم رقابت استعمار) نشان داده شده است.



نمودار ٤-٣ (همگرایی الگوریتم رقابت استعماری بکارگرفته شده در مثال ١)



نمودار ٤-٤(همگرایی الگوریتم رقابت استعماری بکارگرفته شده در مثال ۲)

همانطور که در نمودار ٤-۳ (همگرایی الگوریتم رقابت استعماری بکارگرفته شده در مثال ۱ ونمودار ٤-٤(همگرایی الگوریتم رقابت استعماری بکارگرفته شده در مثال ۲ قابل مشاهده می‌باشد در مثال اول الگوریتم قبل از تکرار دهم و در مثال دوم قبل از تکرار پانزدهم به نقطه بهینه همگرا شده‌اند.

شیوه دیگر نشان دادن همگرایی الگوریتم پیشنهادی که در ادبیات هم متداول است تکرار الگوریتم به اندازه زیاد و مشاهده میزان انحراف آن است.

برای دو مثال ، ۱۰۰ مرتبه با جواب اولیه متفاوت، الگوریتم پیشنهادی رقابت استعماری اجرا شده که در پیوست مقادیر آن آورده شده است. و نتایج حاصل از تحلیل آن در جدول زیر نشان شده است. لازم به توضیح است که نتایج بدست آمده در جدول ٤-۳ (نتایج حل مثال ۱ توسط الگوریتم رقابت استعماری) و جدول ٤-٤(نتایج حل مثال ۲ توسط الگوریتم رقابت استعمار)، حاصل بهترین نتیجه از ۱۰۰ تکرار است.

جدول ٤-٥ (نتایج آماری بدست آمده از ۱۰۰ تکرار الگوریتم پیشنهادی برای دو مثال)

Example	Max	Min	Average	Standard deviation
1	*0/9235*	*0/9036*	*0/920605*	*0/0038265*
2	*0/6571*	*0/6043*	*0/64031*	*0.011437*

همانطور که در جدول بالا نشان داده شده است انحراف معیار خیلی کمی برای دو مثال بدست آمده است که این خود حاکی از تمرکز این الگوریتم روی نقطه نزدیک به بهینه است.

فصـــــل پنجم

نتیـــجه گیری و پیـــشنهادات

۵-۱. مقدمه

دسترسی پذیری یکی از معیارهای عملکردی سیستم‌های تعمیرپذیر است. با توجه به اهمیت این مسأله در دنیای واقعی، برای افزایش مقدار این معیار عملکردی در سیستم‌ها راهکارهای مختلفی توسط پژوهش‌گران پیشنهاد شده است. از مهم‌ترین آنها تخصیص افزونگی است. در این پژوهش با ترکیب دو ساختار سری-موازی و ساختار k از n، مسأله تخصیص افزونگی برای این ساختار ترکیبی مدنظر قرار داده شده است. نکته قابل توجه در این مدل اضافه شدن متغیر تعداد تعمیرکاران در کنار تعداد سطح افزونگی در هر زیر سیستم تحت محدودیت‌های هزینه‌ای، وزنی و حجمی است. با استفاده از فرآیند زنجیره مارکوفی، مدل ایجاد شده از نوع برنامه‌ریزی غیر خطی عدد صحیح با محدودیت‌های خطی است که برای حل آن از الگوریتم فرا ابتکاری رقابت استعماری استفاده شده است.

۵-۲. نتیجه‌گیری

مدل استفاده شده در این پژوهش، از نوع غیرخطی عددصحیح بوده و به مراتب نقاط در فضای جواب گسسته-اند. با توجه به محدودیت نرم افزارهای موجود در بازار برای حل تمامی مسائل غیرخطی، مدل ایجاد شده در پژوهش جزو آن دسته از مسائلی بوده است که توسط نرم افزارهای همچون GAMS قابل حل نبوده است.

برای مقایسه و نشان دادن کارایی الگوریتم رقابت استعماری پیشنهادی چاره‌ای جز پیدا کردن جواب‌های دقیق مثال‌های مدل را نداشتیم. بدین منظور با استفاده از نرم افزار متلب کد مسائل بگونه‌ای نوشته شده است که تمامی نقاط در فضای جواب در تابع هدف مسأله قرار گرفته و تمامی جواب‌ها ذخیره می‌شوند و از بین تمامی جواب-های ذخیره شده بهترین جواب، جواب مسأله ما می‌باشد. چیزی که دقیق بودن این الگوریتم را تضمین می‌کند جستجوی تمامی نقاط در فضای جواب است.

اما چون این روش بدلیل محدودیت‌های زمانی و هزینه‌ای برای مسائل بزرگ بصرفه و منطقی نیست از الگوریتم رقابت استعماری برای حل این مدل استفاده شده است. برای نشان دادن کارایی این الگوریتم از دو مثال استفاده شده است. با توجه به مدل مسأله یکی از فاکتورهایی که موجب افزایش فضای جواب مسأله است افزایش زیرسیستم‌ها می‌باشد، زیرا با افزایش هر زیر سیستم به بعد مسأله یک واحد افزوده می‌شود.

مثال اول با سه زیرسیستم، تقریبا ۱.۲۸ ساعت، زمان برای پیدا کردن حل دقیق توسط نرم افزار متلب صرف شده است. و با افزایش بعد مسأله به چهار زیر سیستم در مثال دوم زمان صرف شده در حدود ۱٤٤.۸ ساعت می-

باشد. و زمان تخمینی برای مسأله‌ای با پنج زیرسیستم حدود ۱۶۳۸۹ ساعت محاسبه شده است که تقریبا امکان پذیر نمی‌باشد.

اما زمان صرف شده برای بدست آوردن جواب نزدیک به بهینه با استفاده از الگوریتم رقابت استعماری برای هر دو مثال کمتر از ۱ دقیقه بدست آمده است. این مسأله به تنهایی حاکی از آن است که اگر روشی استفاده شود که نیاز به زمانی خیلی کمتری داشته باشد کارا است.

کارا بودن الگوریتم رقابت استعماری از دو جهت بررسی شده است؛ یکی بصرفه بودن از نظر زمانی و دیگر همگرایی این الگوریتم بوده است.

این الگوریتم بعد از چندین تکرار با توجه به نمودار ٤-۳ (همگرایی الگوریتم رقابت استعماری بکارگرفته شده در مثال ۱ و نمودار ٤-٤(همگرایی الگوریتم رقابت استعماری بکارگرفته شده در مثال ۲ به نقطه بهینه همگرا است. و با توجه به جدول ٤-۵ (نتایج آماری بدست آمده از ۱۰۰ تکرار الگوریتم پیشنهادی برای دو مثال)، هر دومثال ۱۰۰ مرتبه توسط الگوریتم رقابت استعماری پیشنهادی حل شده است، و انحراف استاندارد ۱۰۰ مرتبه حل در حدود صدم است که مقدار خیلی کوچکی است و همچنین اختلاف میان بهترین و بدترین جواب برای هر دو مثال زیر صدم است.

با توجه به مطالب بیان شده الگوریتم رقابت استعماری پیشنهادی، الگوریتم مناسبی برای حل این نوع مسأله است.

۵-۳. پاسخ به سوالات تحقیق

در این قسمت سوالات تحقیق مجددا مورد بررسی قرار گرفته و با توجه به کل تحقیق و مدل ارائه شده، این سوالات به شرح زیر پاسخ داده می‌شود:

سوال اصلی: مدل مناسب سیستم سری-موازی با اجزای افزونه جهت دستیابی به حداکثر دسترس‌پذیری با توجه به محدودیت‌های حجمی، وزنی و هزینه‌ای چگونه است؟

مدل ریاضی این مساله به کمک مفاهیم مدل سازی ریاضی، زنجیره مارکف، سیستم‌های سری-موازی و سیستم‌های k از n ساخته شده است.و مدل ساخته شده از نوع غیرخطی عدد صحیح بوده و به مراتب نقاط در فضای جواب گسسته‌اند.

سوال فرعی اول: آیا الگوریتم رقابت استعماری می‌تواند الگوریتم مناسبی برای حل مسائل NP-hard باشد؟

همانطور که در فصول قبل بررسی شده این نوع مسائل از دسته مسائل Np-hard می‌باشد و چاره‌ای جز استفاده از الگوریتم‌های فرا ابتکاری برای حل آن باقی نمی‌ماند و با توجه به نتایج بدست آمده الگوریتم رقابت استعماری پیشنهادی، الگوریتم مناسبی برای حل این نوع مسأله است.

سوال فرعی دوم: نقصان روش‌های دیگر حل مانند حل دقیقه مسأله چیست؟

با توجه به محدودیت نرم افزارهای موجود در بازار برای حل تمامی مسائل غیرخطی، مدل ایجاد شده در پژوهش جزو آن دسته از مسائلی بوده که توسط نرم افزارهای همچون GAMS قابل حل نمی‌باشد. و حل دقیق پیشنهادی هم بدلیل جستجوی تمامی نقاط در فضای جواب بسیار زمانبر بوده و با افزایش ابعاد مساله دیگر قادر به پیدا کردن جواب نمی‌باشد و یا باید هزینه و زمان زیادی برای آن صرف کرد.

۵-٤. میزان دستیابی به اهداف تحقیق

یکی از اهداف این تحقیق ارائه یک مدل جهت دستیابی به حداکثر دسترسی پذیری سیستم سری- موازی با توجه به تعمیر پذیر بودن اجزاء و محدویت‌های حجمی، وزنی و هزینه‌ای بوده که این مهم با کمک زنجیره مارکف، مدل سازی ریاضی، قوانین حاکم بر شبکه‌های سری-موازی و سیستم‌های k از n به انجام رسیده است.

و هدف دیگر استفاده از الگوریتم رقابت استعماری برای بدست آوردن متغیرهای بهینه سیستم(تعداد تعمیرکاران و تعداد اجزای افزونه هر زیر سیستم و همچنین ماکزیمم دسترس پذیری) می‌باشد. همانطور که در فصل چهارم نشان داده شد، این الگوریتم بسیار کارا برای بدست آوردن متغیرها بوده است.

۵-۵. پیشنهادات آتی

درپایان لازم به ذکر است که هنوز جای تحقیق و توسعه بسیاری بر روی مدل مطرح شده وجود دارد تا آنرا به مدلی برای حل یک نمونه کاملا واقعی تبدیل نماید، برای این کار می‌توان بسیاری از محدودیت‌ها و فرضیات مهمی را که در دنیای واقعی با آن مواجه می‌شویم مانند در نظر گرفتن حالت‌های مختلف خرابی اجزاء سیستم، بررسی انواع هزینه‌ها و دیگر محدودیت‌ها را در تعریف مسأله لحاظ نمود.

همچنین می‌توان این مدل را برای ساختارهای دیگر سیستم‌های تعمیر پذیر همچون سیستم‌های موازی–سری، سری–موازی و انواع سیستم‌های ترکیبی بکار برد.

پیشنهاد دیگر برای این مسأله استفاده از انواع توزیع‌های آماری همچون توزیع گاما برای توزیع طول عمر اجزاء سیستم و زمان تعمیر بکار برد.

و در آخر استفاده از الگوریتم‌های ابتکاری دیگر همچون ژنتیک، مورچگان و ... برای حل انواع این مدل‌ها و مقایسه الگوریتم‌ها با الگوریتم پیشنهادی می‌تواند از دیگر تحقیقات آتی برای این پژوهش باشد.

منـــــابع

٦-١. مقاله

[1]. Barron, Y., Frostig, E., and Levikson, B. (2006), "Analysis of r out of n systems with several repairmen, exponential life times and phase type repair times: an algorithmic approach", Eur J Oper Res Algorithmic Approach , 169:202–25.

[2] Bulfin, R. L., and Liu, C. Y., (1985), "Optimal allocation of redundant components for large systems", IEEE Transactions on Reliability, 34, 241–248.

[3] Chern, M.S., (1992), "On the computational complexity of reliability redundancy allocation in a series system", Oper Res Lett , 11:309–15.

[4] Chen, T.C., (2006), "IAs based approach for reliability redundancy allocation problems" Applied Mathematics And Computation, 182:1556–1567.

[5] Coit, D.W., and Smith, A., (1996), "Reliability optimization of series-parallel systems using a genetic algorithm", IEEE Trans Reliab, 45(2):254–60.

[6]. Coit, W. D., and JIACHEN, L., (2000), "SYSTEM RELIABILITY OPTIMIZATION WITH k-out-of-n SUBSYSTEMS", International Journal of Reliability, Quality and Safety Engineering, 7(2): 129-142.

[7] Fawzi, B.B., and Hawkes, A.G., "Availability of an r-out-of-n system with spares and repairs" J Appl Probab 1991;28:397–408.

[8] Fyffe, D.E.; Hines, W.W., and Lee, N.K., (1968), "System reliability allocation and a computational algorithm", IEEE Transactions on Reliability, 17:64-69.

[9] Gen, M., and Yun, Y.S., "Soft computing approach for reliability optimization: state-of-the-art survey", Reliab Eng Syst Safety 2006;91(9):1008–26.

[10] Goel, L.R., R. Gupta, R., Gupta, P., "A single unit multi-component system subject to various types of failures", Microelectronics and Reliability 23 (1983) 813–816.

[11] Gupta, P., Lal, A., Sharma, R., and Singh, J., "Numerical analysis of reliability and availability of the series processes in butter oil processing plant", International Journal of Quality & Reliability Management, vol.22, no. 3, pp. 303-16, 2005.

[12] Karin, S., Matthieu, C., Heijden, V.D., and Harten, A.V., (2007), "Availability of k-out-of-N systems under block replacement sharing limited spares and repair capacity" Int. J. Production Economics 107: 404–421.

[13] Kuo, W., Prasad, V.R., Tillman, F.A., and Hawang, C., "Optimal reliability design fundamental and application" London: Cambridge University Press; 2001.

[14] Kennedy, J., and Eberhart, R.C., (1997), "A discrete binary version of the particle swarm algorithm", Proc. 1997 Conf. on Systems, Manand Cybernetics,4104-4109. Piscataway,NJ: IEEE Service Center.

[15] Linmin, H., Dequan, Y., and Jiandang, L., (2012), "Availability analysis and desing optimization for a repairable series-parallel system with failure dependencies", international journal of innovation computing, information and control, volume 8, Number 10(A).

[16] Li, X., Zuo, M.J., and Yam, R.C.M., "Reliability analysis of a repairable k-out-of-n system with some components being suspended when the system is down", Reliability Engineering and System Safety (2006), 91:305-310.

[17] Misra, K.B., and Sharma, U., (1991), "An efficient algorithm to solve integer programming problems arising in system reliability design", IEEE Transactions on Reliability, 40(1), 81–91.

[18] Mo, Y., Amari, S., and Dugan, J.B., "Efficient analysis of multi-state k-out-of-n systems" Original research article reliability Engineering & system volume 133, January 2015,pages 95-105.

[19] Moustafa, M.S., (1996), "Transient analysis of reliability with and without repair for k-out-of-n:G system with two failures modes", Reliability Engineering and System Safety, 53:31-35.

[20] Sadan, K.K., Alice, E.S., and David W.C., (2003), "Efficiently solving the redundancy allocation problem using tabu search", IIE Transactions 35: 515–526.

[21]Smit-Destombes, K.S.D., Heijden, M.C.V., and Harten, A.V., "Availability of k-out-of-n systems under block replacement sharing limited spares and repair capacity" international journal of production economic, Int.j.production Economics 1.7, (2007) 404-21.

[22] Van der Heijden, M, Van Harten, A., Ebben, M., (2001), "Waiting times at periodically switched one-way traffic lanes" Probability in the Engineering and Informational Sciences 15 (4), 495–518.

[23] Wu, W., Tang, Y., Yu, M., and Jian, Y., " Reliability analysis of a k-out-of-n:G repairable system with single vacation" original Research Article Applied Mathematical Modelling, Volum 38,Iss 24, 15 December 2014, pages 6075-6097.

[24] Xie, M., Horigone, M., and Zhang, T., "Availability and reliability of a k-out-of-(M+N):G warm-standby systems" Reliability Engineering and System Safety 91 (2006) 381–387.

[25] Juang, Y.S., Lin, S.S., and Kao, H.P., (2008), "A knowledge management system for series-parallel availability optimization and design", Expert Systems with Applications 34 :181–193.

٦-٢. کتاب

٢٦. توکلی مقدم، ر.، کلامی، س.م.، نوروزی، ن.، و سلامت بخش، ع.، (١٣٩٢)، الگوریتم‌های فراابتکاری مبانی نظری و پیاده سازی در متلب، چاپ اول انتشارات دانشگاه آزاد اسلامی-واحد تهران جنوب،تهران.

٢٧. شریفی، س.م.م.، غلامی مزینان، ح.، کرباسیان، م.، و شریفی، س.م.ح. (١٣٩١)،مهندسی قابلیت اطمینان، چاپ اول انتشارات امید انقلاب، تهران.

٢٨. رضائیان، م.،(١٣٩٣)، ارزیابی قابلیت اطمینان سیستم‌های مهندسی(مفاهیم و روش‌ها)، چاپ پنجم انتشارات دانشگاه صنعتی امیر کبیر(پلی تکنیک تهران).

٢٩. مدرس، م.، و تیموری، ا.، (١٣٩٢)، نظریه صف، چاپ دوم انتشارات داتنشگاه علم و صنعت ایران، تهران.

[30]Ushakov, I., (2012), "Probabilististic reliability models" ISBN:978-118-34183-4.

[31] Amari, S., (2008), " handbook of performability Engineering".

٦-٣. پایان نامه و گزارش‌های علمی

۳۲. آتش پز گرگری، ا. (۱۳۸۷)، " توسعه الگوریتم بهینه سازی اجتماعی و بررسی کارآیی آن". پایان نامه دوره کارشناسی ارشد، دانشگاه تهران.

۳۳. امیری، م. (۱۳۸۵)، "توسعه سیستم‌های پایایی با اجزاء تعمیر پذیر". پایان نامه دوره دکتری، دانشگاه صنعتی شریف.

۳٤. کرد، ع. (۱۳۹۰)، " انتخاب و بهینه سازی سبد سهام با استفاده از روش‌های فرا ابتکاری". پایان نامه دوره کارشناسی ارشد، دانشگاه شریف.

۳۵. یحی تبار عربی، ع.ا. (۱۳۹۱)، "بهینه سازی دسترس پذیری سیستم‌های k از n تعمیر پذیر با الگوریتم pso". پایان نامه دوره کارشناسی ارشد، دانشگاه شریف.

٦-٤. مجموعه مقالات همایش‌ها

۳٦. فارضیان، و.، کرباسیان، م.، و عابدی، س.،(۱۳۹۰)،" ارئه مدلی برای بهینه سازی قابلیت اطمینان در سیستم سری-موازی برمبنای مکانیابی تسهیلات"، دومین کنفرانس مهندسی قابلیت اطمینان، ۲-٤ آبان ۱۳۹۰.

[۳۷] Sharma, S.P., Kumar, D., and Kumar, A., (2008), " Availability optimization series parallel systems using genetic algorithms", XXXII NATIONAL SYSTEMS CONFERENCE, NSC 2008, December 17-19.

٦-٥. منابع الکترونیکی

[38] http://fa.wikipedia.org/wiki
[39] http://www.ronamax.com
[40] http://ssd2012aut.ac.ir

پیـــــوســـــت

۷-۱. کد حل دقیق مثال ۱ توسط نرم افزار متلب

```
clear
clc
xx=0;
tic
y=3;
lambda=1.4;
mu=1.1;
k=[ 2 1 2 ];
n=[ 0 0 0 ];
r=[ 0 0 0 ];
c=[ 5 4 7 ];
h=[ 1 1 2 ];
w=[ 0.3 0.4 0.27 ];
v=[ 1.1 1.3 2.1 ];
for b=2:40
    n(1,3)=0;
    n(1,3)=n(1,1)+b;
    n(1,2)=0;
    n(1,1)=0;
    for j=1:40
        n(1,2)=0;
        n(1,2)=n(1,2)+j;
        n(1,1)=0;
        for i=2:40
            n(1,1)=0;
            n(1,1)=n(1,1)+i;
            for z=1:n(3)-k(3)+1
                r(1,3)=0;
                r(1,3)=r(1,3)+z;
                r(1,2)=0;
                r(1,1)=0;
                for t=1:n(2)-k(2)+1
                    r(1,2)=0;
                   r(1,2)=r(1,2)+t;
                    r(1,1)=0;
                    for q=1:n(1)-k(1)+1
                        r(1,1)=0;
                        r(1,1)=r(1,1)+q;
                        CC=sum(c.*n)+sum(r.*h);
                        WW=sum(w.*n);
                        VV=sum(v.*n);
                        if(CC<=480 && WW<=45 && VV<=40)
                            xx=xx+1;
                            As=1;
                            for jj=1:y
                               f1=0;
                                for ii=1:r(jj)
```

```
                                f1=factorial(n(jj))/factorial(ii)/factorial(n(jj)-
ii)*(lambda/mu)^ii+f1;
                                    end
                                    f2=0;
                                    for ii=r(jj)+1:n(jj)-k(jj)+1
                            f2=factorial(n(jj))/factorial(r(jj))/factorial(n(jj)-
ii)*(lambda/mu)^ii/(r(jj)^(ii-r(jj)))+f2;
                                    end
                                            f3=factorial(n(jj))/(factorial(k(jj)-
1)*factorial(r(jj))*r(jj)^(n(jj)-k(jj)-r(jj)+1))*(lambda/mu)^(n(jj)-k(jj)+1);
                                  As=(1-f3/(1+f1+f2))*As;
                                    ASGROUP(xx)=As;
                                    PARAMETERGROUP(xx,:)=[n r];
                                end
                            end
                        end
                    end
                end
            end
        end
    end
    toc
    [FINAL_AS, AS_NO]=max(ASGROUP);
    FINAL_PARAMETER=PARAMETERGROUP(AS_NO,:);
    display(FINAL_AS);
    display(FINAL_PARAMETER);
    save result PARAMETERGROUP ASGROUP FINAL_AS FINAL_PARAMETER
```

۷-۲. کد حل دقیق مثال ۲ توسط نرم افزار متلب

```
clear
clc
xx=0;
tic
y=4;
lambda=1.4;
mu=1.1;
k=[2 1 2 3];
n=[0 0 0 0];
r=[0 0 0 0];
c=[5 4 7 6];
h=[1 1 2 2];
w=[0.3 0.4 0.27 0.7];
v=[1.1 1.3 2.1 3.1];
for a=3:18
    n(1,4)=0;
    n(1,4)=n(1,4)+a;
    n(1,3)=0;
    n(1,2)=0;
    n(1,1)=0;
 for b=2:27
     n(1,3)=0;
     n(1,3)=n(1,3)+b;
     n(1,2)=0;
     n(1,1)=0;
    for j=1:43
        n(1,2)=0;
        n(1,2)=n(1,2)+j;
        n(1,1)=0;
        for i=2:50
            n(1,1)=0;
            n(1,1)=n(1,1)+i;
          for s=1:n(4)-k(4)+1
              r(1,4)=0;
              r(1,4)=r(1,4)+s;
              r(1,3)=0;
              r(1,2)=0;
              r(1,1)=0;
            for z=1:n(3)-k(3)+1
                r(1,3)=0;
                r(1,3)=r(1,3)+z;
                r(1,2)=0;
                r(1,1)=0;
                for t=1:n(2)-k(2)+1
                    r(1,2)=0;
                    r(1,2)=r(1,2)+t;
                    r(1,1)=0;
```

```
                    for q=1:n(1)-k(1)+1
                        r(1,1)=0;
                        r(1,1)=r(1,1)+q;
                        CC=sum(c.*n)+sum(r.*h);
                        WW=sum(w.*n);
                        VV=sum(v.*n);
                        if(CC<=261 && WW<=35 && VV<=55)
                            xx=xx+1;
                            As=1;
                            for jj=1:y
                                f1=0;
                                for ii=1:r(jj)

f1=factorial(n(jj))/factorial(ii)/factorial(n(jj)-ii)*(lambda/mu)^ii+f1;
                                end
                                f2=0;
                                for ii=r(jj)+1:n(jj)-k(jj)+1

f2=factorial(n(jj))/factorial(r(jj))/factorial(n(jj)-
ii)*(lambda/mu)^ii/(r(jj)^(ii-r(jj)))+f2;
                                end
                                f3=factorial(n(jj))/(factorial(k(jj)-
1)*factorial(r(jj))*r(jj)^(n(jj)-k(jj)-r(jj)+1))*(lambda/mu)^(n(jj)-k(jj)+1);
                                As=(1-f3/(1+f1+f2))*As;
                                ASGROUP(xx)=As;
                                PARAMETERGROUP(xx,:)=[n r];
                            end
                        end
                    end
                end
            end
        end
    end
  end
  toc
  end
  [FINAL_AS, AS_NO]=max(ASGROUP);
  FINAL_PARAMETER=PARAMETERGROUP(AS_NO,:);
  display(FINAL_AS);
  display(FINAL_PARAMETER);
  save result PARAMETERGROUP ASGROUP FINAL_AS FINAL_PARAMETER
```

۷-۳. کد الگوریتم رقابت استعماری برای مثال ۱

ica.m . ۱-۳-۷

```
clc;
clear;
close all;
tic;
%% Problem Definition

CostFunction=@(x) cost1(x);        % Cost Function

nVar=6;             % Number of Decision Variables

VarSize=[1 nVar];   % Decision Variables Matrix Size

VarMin=3;          % Lower Bound of Variables
VarMax=13;          % Upper Bound of Variables


%% ICA Parameters

MaxIt=100;          % Maximum Number of Iterations

nPop=70;            % Population Size
nEmp=10;            % Number of Empires/Imperialists

alpha=1;            % Selection Pressure

beta=2;             % Assimilation Coefficient

pRevolution=0.1;    % Revolution Probability
mu=0.05;            % Revolution Rate

zeta=0.1;           % Colonies Mean Cost Coefficient

ShareSettings;


%% Initialization

% Initialize Empires
emp=CreateInitialEmpires();

% Array to Hold Best Cost Values
BestCost=zeros(MaxIt,1);


%% ICA Main Loop

for it=1:MaxIt

    % Assimilation
```

```
    emp=AssimilateColonies(emp);

    % Revolution
    emp=DoRevolution(emp);

    % Intra-Empire Competition
    emp=IntraEmpireCompetition(emp);

    % Update Total Cost of Empires
    emp=UpdateTotalCost(emp);

    % Inter-Empire Competition
    emp=InterEmpireCompetition(emp);

    % Update Best Solution Ever Found
    imp=[emp.Imp];
    [~, BestImpIndex]=min([imp.Cost]);
    BestSol=imp(BestImpIndex);

    % Update Best Cost
    if it==1
     if BestSol.Cost>0
     BestCost(it)=0;
        BesttSoll{it}=round(BestSol.Position);

    end
    if BestSol.Cost<0
    BestCost(it)=BestSol.Cost;
        BesttSoll{it}=round(BestSol.Position);

    end
    end
    if it>1
     if BestSol.Cost>0
     BestCost(it)=0;
         BesttSoll{it}=round(BestSol.Position);

    end
    if BestSol.Cost<0 && BestSol.Cost<=BestCost(it-1)
    BestCost(it)=BestSol.Cost;
        BesttSoll{it}=round(BestSol.Position);

    end
    if BestSol.Cost<0 && BestSol.Cost>BestCost(it-1)
    BestCost(it)=BestCost(it-1);
        BesttSoll{it}=BesttSoll{it-1};

    end
    end


    % Show Iteration Information
    disp(['Iteration ' num2str(it) ', SOL: ' num2str(BesttSoll{it}) ',Best Cost
= ' num2str(-BestCost(it))]);
    if it==MaxIt
       n = BesttSoll{it}(1:3)
       r = BesttSoll{it}(4:6)
```

```
        BEST_Reliability=-BestCost(it)
    end
end

%% Results

figure;
plot(-BestCost,'LineWidth',2.5);
xlabel('Iteration');
ylabel('Best Cost');
toc;
```

cost1.m . ۲-۳-۷

```
function z= cost1(x)

j=3;
xx=round(x);
n=xx(1:3);
r=xx(4:6);
excep=0;
for ii=1:j

    if n(ii)<r(ii)
        excep=excep+1;
    end
    if n(ii)<=2
     excep=excep+1;
    end
    if r(ii)<=1
     excep=excep+1;
    end

end
if excep==0

k=[2 1 2];
c=[5 4 7];
h=[1 1 2];
v=[1.1 1.3 2.1];
w=[0.3 0.4 0.27];
W=45;
V=40;
C=480;
Y=3;
mu=1.1;
landa=1.4;
 penalty1=0;


for ii=1:j

   if n(ii)<k(ii)
       penalty1=penalty1+0.5;
   end
```

```
end
penalty2=0;
for ii=1:j

 if r(ii)>(n(ii)-k(ii)+1) || r(ii)<1
     penalty2=penalty2+0.5;

 end

end
c11=0;
for ii=1:j
c11=c11+((c(ii)*n(ii))+(r(ii)*h(ii)));

end

if c11>C
    penalty3=0.5;
else penalty3=0;

end
c12=0;
for ii=1:j
   c12=c12+(w(ii)*n(ii));

end
if c12>W
    penalty4=0.5;
else
    penalty4=0;
end
c13=0;
for ii=1:j

   c13=c13+(v(ii)*n(ii));

end
if c13>V
    penalty5=0.5;
else
    penalty5=0;
end
f1=0;
f2=0;
ff=1;
for j=1:Y

 for i=1:r(j)

     f1=f1+((factorial(n(j))/(factorial(i)*factorial(n(j)-i)))*((landa/mu)^i));
 end
 if  (r(j)+1)<=(n(j)-k(j)+1)
 for i=(r(j)+1):(n(j)-k(j)+1)
    f2=f2+((factorial(n(j))*((landa/mu)^i))/((factorial(n(j)-
i))*(factorial(r(j)))*(r(j)^(i-r(j)))));
```

```
 end
 else
     f2=0;
 end

 f3=(((factorial(n(j)))*((landa/mu)^(n(j)-k(j)+1)))/((factorial(k(j)-
1))*(factorial(r(j)))*(r(j)^(n(j)-k(j)-r(j)+1))));

 ff=ff*(1-((1/(1+f1+f2))*(f3)));
 f1=0;
 f2=0;
 f3=0;
end
PENLATY=penalty1+penalty2+penalty3+penalty4+penalty5;
z=-ff+PENLATY;
else
    z=0.5;
end
end
```

sharesettings.m . ٣-٣-٧

```
global ProblemSettings;
ProblemSettings.CostFunction=CostFunction;
ProblemSettings.nVar=nVar;
ProblemSettings.VarSize=VarSize;
ProblemSettings.VarMin=VarMin;
ProblemSettings.VarMax=VarMax;

global ICASettings;
ICASettings.MaxIt=MaxIt;
ICASettings.nPop=nPop;
ICASettings.nEmp=nEmp;
ICASettings.alpha=alpha;
ICASettings.beta=beta;
ICASettings.pRevolution=pRevolution;
ICASettings.mu=mu;
ICASettings.zeta=zeta;
```

.CreateInitialEmpires.m . ٤-٣-٧

```
function emp=CreateInitialEmpires()

    global ProblemSettings;
    global ICASettings;

    CostFunction=ProblemSettings.CostFunction;
    nVar=ProblemSettings.nVar;
    VarSize=ProblemSettings.VarSize;
    VarMin=ProblemSettings.VarMin;
    VarMax=ProblemSettings.VarMax;

    nPop=ICASettings.nPop;
    nEmp=ICASettings.nEmp;
    nCol=nPop-nEmp;
    alpha=ICASettings.alpha;
```

```
empty_country.Position=[];
empty_country.Cost=[];
kk=[2 1 2];

country=repmat(empty_country,nPop,1);

for i=1:nPop
    dd=randi([VarMin,VarMax],1,nVar-3);
for u=1:3
 dd1(u)=randi([1,(dd(u)-kk(u)+1)],1,1);

end

    country(i).Position=[dd dd1];

    country(i).Cost=CostFunction(country(i).Position);

end


costs=[country.Cost];
[~, SortOrder]=sort(costs);

country=country(SortOrder);

imp=country(1:nEmp);

col=country(nEmp+1:end);


empty_empire.Imp=[];
empty_empire.Col=repmat(empty_country,0,1);
empty_empire.nCol=0;
empty_empire.TotalCost=[];

emp=repmat(empty_empire,nEmp,1);

% Assign Imperialists
for k=1:nEmp
    emp(k).Imp=imp(k);
end

% Assign Colonies
P=exp(-alpha*[imp.Cost]/max([imp.Cost]));
P=P/sum(P);
for j=1:nCol

    k=RouletteWheelSelection(P);

    emp(k).Col=[emp(k).Col
                col(j)];

    emp(k).nCol=emp(k).nCol+1;
end
```

```
    emp=UpdateTotalCost(emp);

end
```

AssimilateColonies.m . ٥-٣-٧

```
function emp=AssimilateColonies(emp)

    global ProblemSettings;
    CostFunction=ProblemSettings.CostFunction;
    VarSize=ProblemSettings.VarSize;
    VarMin=ProblemSettings.VarMin;
    VarMax=ProblemSettings.VarMax;

    global ICASettings;
    beta=ICASettings.beta;

    nEmp=numel(emp);
    for k=1:nEmp
        for i=1:emp(k).nCol

            emp(k).Col(i).Position = emp(k).Col(i).Position ...
                + beta*rand(VarSize).*(emp(k).Imp.Position-
emp(k).Col(i).Position);

            emp(k).Col(i).Position = max(emp(k).Col(i).Position,VarMin);
            emp(k).Col(i).Position = min(emp(k).Col(i).Position,VarMax);

            emp(k).Col(i).Cost = CostFunction(emp(k).Col(i).Position);

        end
    end

end
```

DoRevolution.m . ٦-٣-٧

```
function emp=DoRevolution(emp)

    global ProblemSettings;
    CostFunction=ProblemSettings.CostFunction;
    nVar=ProblemSettings.nVar;
    VarSize=ProblemSettings.VarSize;
    VarMin=ProblemSettings.VarMin;
    VarMax=ProblemSettings.VarMax;

    global ICASettings;
    pRevolution=ICASettings.pRevolution;
    mu=ICASettings.mu;

    nmu=ceil(mu*nVar);

    sigma=0.1*(VarMax-VarMin);

    nEmp=numel(emp);
```

```
    for k=1:nEmp

        NewPos = emp(k).Imp.Position + sigma*randn(VarSize);

        jj=randsample(nVar,nmu)';
        NewImp=emp(k).Imp;
        NewImp.Position(jj)=NewPos(jj);
        NewImp.Cost=CostFunction(NewImp.Position);
        if NewImp.Cost<emp(k).Imp.Cost
            emp(k).Imp = NewImp;
        end

        for i=1:emp(k).nCol
            if rand<=pRevolution

                NewPos = emp(k).Col(i).Position + sigma*randn(VarSize);

                jj=randsample(nVar,nmu)';
                emp(k).Col(i).Position(jj) = NewPos(jj);

                emp(k).Col(i).Position = max(emp(k).Col(i).Position,VarMin);
                emp(k).Col(i).Position = min(emp(k).Col(i).Position,VarMax);

                emp(k).Col(i).Cost = CostFunction(emp(k).Col(i).Position);

            end
        end
    end

end
```

IntraEmpireCompetition.m . ۷–۳–۷

```
function emp=IntraEmpireCompetition(emp)

    nEmp=numel(emp);

    for k=1:nEmp
        for i=1:emp(k).nCol
            if emp(k).Col(i).Cost<emp(k).Imp.Cost
                imp=emp(k).Imp;
                col=emp(k).Col(i);

                emp(k).Imp=col;
                emp(k).Col(i)=imp;
            end
        end
    end

end
```

UpdateTotalCost.m . ۸–۳–۷

```
function emp=UpdateTotalCost(emp)
```

```
    global ICASettings;
    zeta=ICASettings.zeta;

    nEmp=numel(emp);

    for k=1:nEmp
        if emp(k).nCol>0
            emp(k).TotalCost=emp(k).Imp.Cost+zeta*mean([emp(k).Col.Cost]);
        else
            emp(k).TotalCost=emp(k).Imp.Cost;
        end
    end

end
```

InterEmpireCompetition.m . ۹-۳-۷

```
function emp=InterEmpireCompetition(emp)

    if numel(emp)==1
        return;
    end

    global ICASettings;
    alpha=ICASettings.alpha;

    TotalCost=[emp.TotalCost];

    [~, WeakestEmpIndex]=max(TotalCost);
    WeakestEmp=emp(WeakestEmpIndex);

    P=exp(-alpha*TotalCost/max(TotalCost));
    P(WeakestEmpIndex)=0;
    P=P/sum(P);
    if any(isnan(P))
        P(isnan(P))=0;
        if all(P==0)
            P(:)=1;
        end
        P=P/sum(P);
    end

    if WeakestEmp.nCol>0
        [~, WeakestColIndex]=max([WeakestEmp.Col.Cost]);
        WeakestCol=WeakestEmp.Col(WeakestColIndex);

        WinnerEmpIndex=RouletteWheelSelection(P);
        WinnerEmp=emp(WinnerEmpIndex);

        WinnerEmp.Col(end+1)=WeakestCol;
        WinnerEmp.nCol=WinnerEmp.nCol+1;
        emp(WinnerEmpIndex)=WinnerEmp;

        WeakestEmp.Col(WeakestColIndex)=[];
        WeakestEmp.nCol=WeakestEmp.nCol-1;
        emp(WeakestEmpIndex)=WeakestEmp;
    end
```

```
    if WeakestEmp.nCol==0

        WinnerEmpIndex2=RouletteWheelSelection(P);
        WinnerEmp2=emp(WinnerEmpIndex2);

        WinnerEmp2.Col(end+1)=WeakestEmp.Imp;
        WinnerEmp2.nCol=WinnerEmp2.nCol+1;
        emp(WinnerEmpIndex2)=WinnerEmp2;

        emp(WeakestEmpIndex)=[];
    end

end
```

RouletteWheelSelection.m . ١٠–٣–٧

```
function i=RouletteWheelSelection(P)

    r=rand;

    C=cumsum(P);

    i=find(r<=C,1,'first');

end
```

٧–٤. كد الگوريتم رقابت استعماری برای مثال ٢

ica.m . ١–٤–٧

```
clc;
clear;
close all;
tic;
%% Problem Definition

CostFunction=@(x) cost1(x);        % Cost Function

nVar=8;             % Number of Decision Variables

VarSize=[1 nVar];   % Decision Variables Matrix Size

VarMin=3;          % Lower Bound of Variables
VarMax=10;          % Upper Bound of Variables


%% ICA Parameters

MaxIt=200;          % Maximum Number of Iterations
```

```
nPop=50;            % Population Size
nEmp=10;            % Number of Empires/Imperialists

alpha=1;            % Selection Pressure

beta=2;             % Assimilation Coefficient

pRevolution=0.1;    % Revolution Probability
mu=0.05;            % Revolution Rate

zeta=0.1;           % Colonies Mean Cost Coefficient

ShareSettings;


%% Initialization

% Initialize Empires
emp=CreateInitialEmpires();

% Array to Hold Best Cost Values
BestCost=zeros(MaxIt,1);


%% ICA Main Loop

for it=1:MaxIt

    % Assimilation
    emp=AssimilateColonies(emp);

    % Revolution
    emp=DoRevolution(emp);

    % Intra-Empire Competition
    emp=IntraEmpireCompetition(emp);

    % Update Total Cost of Empires
    emp=UpdateTotalCost(emp);

    % Inter-Empire Competition
    emp=InterEmpireCompetition(emp);

    % Update Best Solution Ever Found
    imp=[emp.Imp];
    [~, BestImpIndex]=min([imp.Cost]);
    BestSol=imp(BestImpIndex);

    % Update Best Cost
    if it==1
     if BestSol.Cost>0
     BestCost(it)=0;
        BesttSoll{it}=round(BestSol.Position);

    end
    if BestSol.Cost<0
```

```
    BestCost(it)=BestSol.Cost;
        BesttSoll{it}=round(BestSol.Position);

    end
    end
    if it>1
     if BestSol.Cost>0
     BestCost(it)=0;
         BesttSoll{it}=round(BestSol.Position);

    end
    if BestSol.Cost<0 && BestSol.Cost<=BestCost(it-1)
    BestCost(it)=BestSol.Cost;
        BesttSoll{it}=round(BestSol.Position);

    end
    if BestSol.Cost<0 && BestSol.Cost>BestCost(it-1)
    BestCost(it)=BestCost(it-1);
        BesttSoll{it}=BesttSoll{it-1};

    end
    end


    % Show Iteration Information
    disp(['Iteration ' num2str(it) ', SOL: ' num2str(BesttSoll{it}) ',Best Cost
= ' num2str(-BestCost(it))]);
    if it==MaxIt
       n = BesttSoll{it}(1:4)
       r = BesttSoll{it}(5:8)
       BEST_Reliability=-BestCost(it)
    end

end

%% Results

%
figure;
plot(-BestCost,'LineWidth',2.5);
xlabel('Iteration');
ylabel('Best Cost');
toc;
```

cost1.m . ٢-٤-٧

```
function z= cost1(x)

j=4;
xx=round(x);
n=xx(1:4);
r=xx(5:8);
excep=0;
for ii=1:j
```

```
    if n(ii)<r(ii)
        excep=excep+1;
    end
    if n(ii)<=2
     excep=excep+1;
    end
    if r(ii)<=1
     excep=excep+1;
    end

end
if excep==0

k=[2 1 2 3];
c=[5 4 7 6];
h=[1 1 2 2];
v=[1.1 1.3 2.1 3.1];
w=[0.3 0.4 0.27 0.7];
W=35;
V=55;
C=261;
Y=4;
mu=1.1;
landa=1.4;


penalty1=0;


for ii=1:j

   if n(ii)<k(ii)
       penalty1=penalty1+0.5;
   end

end
penalty2=0;
for ii=1:j

 if r(ii)>(n(ii)-k(ii)+1) || r(ii)<1
     penalty2=penalty2+0.5;

 end

end
c11=0;
for ii=1:j
c11=c11+((c(ii)*n(ii))+(r(ii)*h(ii)));

end

if c11>C
    penalty3=0.5;
else penalty3=0;

end
c12=0;
```

```
for ii=1:j
   c12=c12+(w(ii)*n(ii));

end
if c12>W
    penalty4=0.5;
else
    penalty4=0;
end
c13=0;
for ii=1:j

   c13=c13+(v(ii)*n(ii));

end
if c13>V
    penalty5=0.5;
else
    penalty5=0;
end
f1=0;
f2=0;
ff=1;
for j=1:Y

 for i=1:r(j)

     f1=f1+((factorial(n(j))/(factorial(i)*factorial(n(j)-i)))*((landa/mu)^i));
 end
  if  (r(j)+1)<=(n(j)-k(j)+1)

 for i=(r(j)+1):(n(j)-k(j)+1)
    f2=f2+((factorial(n(j))*((landa/mu)^i))/((factorial(n(j)-
i))*(factorial(r(j)))*(r(j)^(i-r(j)))));

 end
  else
      f2=0;
  end

 f3=(((factorial(n(j)))*((landa/mu)^(n(j)-k(j)+1)))/((factorial(k(j)-
1))*(factorial(r(j)))*(r(j)^(n(j)-k(j)-r(j)+1))));

 ff=ff*(1-((1/(1+f1+f2))*(f3)));
 f1=0;
 f2=0;
 f3=0;
end
PENLATY=penalty1+penalty2+penalty3+penalty4+penalty5;
z=-ff+PENLATY;
else
    z=0.5;
end
end
```

sharesettings.m . ٣-٤-٧

```
global ProblemSettings;
```

```
ProblemSettings.CostFunction=CostFunction;
ProblemSettings.nVar=nVar;
ProblemSettings.VarSize=VarSize;
ProblemSettings.VarMin=VarMin;
ProblemSettings.VarMax=VarMax;

global ICASettings;
ICASettings.MaxIt=MaxIt;
ICASettings.nPop=nPop;
ICASettings.nEmp=nEmp;
ICASettings.alpha=alpha;
ICASettings.beta=beta;
ICASettings.pRevolution=pRevolution;
ICASettings.mu=mu;
ICASettings.zeta=zeta;
```

.CreateInitialEmpires.m . ٤-٤-٧

```
function emp=CreateInitialEmpires()

    global ProblemSettings;
    global ICASettings;

    CostFunction=ProblemSettings.CostFunction;
    nVar=ProblemSettings.nVar;
    VarSize=ProblemSettings.VarSize;
    VarMin=ProblemSettings.VarMin;
    VarMax=ProblemSettings.VarMax;

    nPop=ICASettings.nPop;
    nEmp=ICASettings.nEmp;
    nCol=nPop-nEmp;
    alpha=ICASettings.alpha;

    empty_country.Position=[];
    empty_country.Cost=[];
    kk=[2 1 2 3];

    country=repmat(empty_country,nPop,1);

    for i=1:nPop
        dd=randi([VarMin,VarMax],1,nVar-4);
    for u=1:4
     dd1(u)=randi([1,(dd(u)-kk(u)+1)],1,1);

    end

        country(i).Position=[dd dd1];

        country(i).Cost=CostFunction(country(i).Position);

    end


    costs=[country.Cost];
```

```
    [~, SortOrder]=sort(costs);

    country=country(SortOrder);

    imp=country(1:nEmp);

    col=country(nEmp+1:end);


    empty_empire.Imp=[];
    empty_empire.Col=repmat(empty_country,0,1);
    empty_empire.nCol=0;
    empty_empire.TotalCost=[];

    emp=repmat(empty_empire,nEmp,1);

    % Assign Imperialists
    for k=1:nEmp
        emp(k).Imp=imp(k);
    end

    % Assign Colonies
    P=exp(-alpha*[imp.Cost]/max([imp.Cost]));
    P=P/sum(P);
    for j=1:nCol

        k=RouletteWheelSelection(P);

        emp(k).Col=[emp(k).Col
                    col(j)];

        emp(k).nCol=emp(k).nCol+1;
    end

    emp=UpdateTotalCost(emp);

end
```

AssimilateColonies.m . ٥-٤-٧

```
function emp=AssimilateColonies(emp)

    global ProblemSettings;
    CostFunction=ProblemSettings.CostFunction;
    VarSize=ProblemSettings.VarSize;
    VarMin=ProblemSettings.VarMin;
    VarMax=ProblemSettings.VarMax;

    global ICASettings;
    beta=ICASettings.beta;

    nEmp=numel(emp);
    for k=1:nEmp
        for i=1:emp(k).nCol
```

```
            emp(k).Col(i).Position = emp(k).Col(i).Position ...
                + beta*rand(VarSize).*(emp(k).Imp.Position-
emp(k).Col(i).Position);

            emp(k).Col(i).Position = max(emp(k).Col(i).Position,VarMin);
            emp(k).Col(i).Position = min(emp(k).Col(i).Position,VarMax);

            emp(k).Col(i).Cost = CostFunction(emp(k).Col(i).Position);

        end
    end

end
```

DoRevolution.m . ٦-٤-٧

```
function emp=DoRevolution(emp)

    global ProblemSettings;
    CostFunction=ProblemSettings.CostFunction;
    nVar=ProblemSettings.nVar;
    VarSize=ProblemSettings.VarSize;
    VarMin=ProblemSettings.VarMin;
    VarMax=ProblemSettings.VarMax;

    global ICASettings;
    pRevolution=ICASettings.pRevolution;
    mu=ICASettings.mu;

    nmu=ceil(mu*nVar);

    sigma=0.1*(VarMax-VarMin);

    nEmp=numel(emp);
    for k=1:nEmp

        NewPos = emp(k).Imp.Position + sigma*randn(VarSize);

        jj=randsample(nVar,nmu)';
        NewImp=emp(k).Imp;
        NewImp.Position(jj)=NewPos(jj);
        NewImp.Cost=CostFunction(NewImp.Position);
        if NewImp.Cost<emp(k).Imp.Cost
            emp(k).Imp = NewImp;
        end

        for i=1:emp(k).nCol
            if rand<=pRevolution

                NewPos = emp(k).Col(i).Position + sigma*randn(VarSize);

                jj=randsample(nVar,nmu)';
                emp(k).Col(i).Position(jj) = NewPos(jj);

                emp(k).Col(i).Position = max(emp(k).Col(i).Position,VarMin);
                emp(k).Col(i).Position = min(emp(k).Col(i).Position,VarMax);
```

```
                emp(k).Col(i).Cost = CostFunction(emp(k).Col(i).Position);

            end
        end
    end

end
```

IntraEmpireCompetition.m . ٧-٤-٧

```
function emp=IntraEmpireCompetition(emp)

    nEmp=numel(emp);

    for k=1:nEmp
        for i=1:emp(k).nCol
            if emp(k).Col(i).Cost<emp(k).Imp.Cost
                imp=emp(k).Imp;
                col=emp(k).Col(i);

                emp(k).Imp=col;
                emp(k).Col(i)=imp;
            end
        end
    end

end
```

UpdateTotalCost.m . ٨-٤-٧

```
function emp=UpdateTotalCost(emp)

    global ICASettings;
    zeta=ICASettings.zeta;

    nEmp=numel(emp);

    for k=1:nEmp
        if emp(k).nCol>0
            emp(k).TotalCost=emp(k).Imp.Cost+zeta*mean([emp(k).Col.Cost]);
        else
            emp(k).TotalCost=emp(k).Imp.Cost;
        end
    end

end
```

InterEmpireCompetition.m . ٩-٤-٧

```
function emp=InterEmpireCompetition(emp)

    if numel(emp)==1
        return;
    end
```

```
    global ICASettings;
    alpha=ICASettings.alpha;

    TotalCost=[emp.TotalCost];

    [~, WeakestEmpIndex]=max(TotalCost);
    WeakestEmp=emp(WeakestEmpIndex);

    P=exp(-alpha*TotalCost/max(TotalCost));
    P(WeakestEmpIndex)=0;
    P=P/sum(P);
    if any(isnan(P))
        P(isnan(P))=0;
        if all(P==0)
            P(:)=1;
        end
        P=P/sum(P);
    end

    if WeakestEmp.nCol>0
        [~, WeakestColIndex]=max([WeakestEmp.Col.Cost]);
        WeakestCol=WeakestEmp.Col(WeakestColIndex);

        WinnerEmpIndex=RouletteWheelSelection(P);
        WinnerEmp=emp(WinnerEmpIndex);

        WinnerEmp.Col(end+1)=WeakestCol;
        WinnerEmp.nCol=WinnerEmp.nCol+1;
        emp(WinnerEmpIndex)=WinnerEmp;

        WeakestEmp.Col(WeakestColIndex)=[];
        WeakestEmp.nCol=WeakestEmp.nCol-1;
        emp(WeakestEmpIndex)=WeakestEmp;
    end

    if WeakestEmp.nCol==0

        WinnerEmpIndex2=RouletteWheelSelection(P);
        WinnerEmp2=emp(WinnerEmpIndex2);

        WinnerEmp2.Col(end+1)=WeakestEmp.Imp;
        WinnerEmp2.nCol=WinnerEmp2.nCol+1;
        emp(WinnerEmpIndex2)=WinnerEmp2;

        emp(WeakestEmpIndex)=[];
    end

end
```

Roulettwheelselection.m . ١٠-٤-٧

```
function i=RouletteWheelSelection(P)

    r=rand;
```

```
    C=cumsum(P);

    i=find(r<=C,1,'first');

end
```

۷-۵. نتایج حاصل از ۱۰۰ بار اجرای مثال ۱ توسط الگوریتم رقابت استعماری

	Elapsed time	n 1	n 2	n 3	r 1	r2	r3	FINAL_AS
1	9.312757	12	6	9	11	6	8	0.9235
2	9.206505	10	6	10	9	6	9	0.9234
3	9.960064	9	7	10	8	7	9	0.9223
4	9.091957	10	9	8	9	9	7	0.9112
5	9.296863	13	5	9	12	5	8	0.9036
6	9.382421	12	6	9	11	6	8	0.9235
7	10.365547	12	6	9	11	6	8	0.9235
8	9.654192	12	6	9	11	6	8	0.9235
9	9.821314	10	9	8	9	8	7	0.9106
10	9.438395	12	6	9	11	6	8	0.9235
11	9.287317	10	6	10	9	6	9	0.9234
12	9.088445	12	6	9	11	6	8	0.9235
13	9.056068	14	6	8	13	6	7	0.9067
14	9.253895	10	9	8	9	9	7	0.9112
15	9.166097	12	6	9	11	6	8	0.9235
16	9.254012	12	6	9	11	6	8	0.9235
17	9.418103	12	7	8	11	7	7	0.9141
18	10.297941	11	8	8	10	8	7	0.9159

19	9.532971	9	8	9	8	8	8	0.9155
20	10.200285	12	6	9	11	6	8	0.9235
21	9.186286	9	7	10	8	7	9	0.9223
22	9.546223	12	6	9	11	6	8	0.9235
23	9.650446	12	6	9	11	6	8	0.9235
24	9.429198	12	6	9	11	6	8	0.9235
25	9.445591	14	6	8	13	6	7	0.9067
26	9.319612	10	7	9	9	7	8	0.9223
27	9.401473	10	7	9	9	7	8	0.9223
28	9.486593	9	7	10	8	7	9	0.9223
29	9.316975	10	9	8	9	9	7	0.9112
30	9.140339	10	6	9	9	6	9	0.9234
31	9.293936	12	6	9	11	5	8	0.9178
32	9.367347	9	7	10	8	7	9	0.9223
33	9.376433	9	8	9	8	8	8	0.9155
34	9.198074	11	8	8	10	8	7	0.9159
35	9.256131	12	6	9	11	6	8	0.9235
36	9.369802	10	6	10	9	6	9	0.9234
37	10.547703	12	6	9	11	6	8	0.9235
38	10.022693	12	6	9	11	6	8	0.9235
39	9.358020	10	6	10	9	6	9	0.9234
40	9.217826	10	9	8	9	9	7	0.9112
41	9.744815	9	8	9	8	8	8	0.9155

42	9.657997	10	6	10	9	6	9	0.9234
43	10.202771	11	8	8	10	8	7	0.9159
44	9.594650	12	6	9	11	6	8	0.9235
45	10.590554	12	7	8	11	7	7	0.9141
46	9.437956	9	7	10	8	7	9	0.9223
47	9.303105	9	7	10	8	7	9	0.9223
48	9.617290	12	6	9	11	6	8	0.9235
49	11.087150	10	7	9	9	7	8	0.9223
50	9.339043	10	9	8	9	9	7	0.9112
51	9.302345	10	7	9	9	7	8	0.9223
52	9.206172	12	6	9	11	6	8	0.9235
53	9.485524	9	8	9	8	8	8	0.9155
54	9.286398	9	7	10	8	7	9	0.9223
55	9.187614	10	7	9	9	7	8	0.9223
56	9.400772	12	6	9	11	6	8	0.9235
57	10.728975	12	6	9	11	6	8	0.9235
58	10.119216	10	6	10	9	6	9	0.9234
59	9.468840	12	6	9	11	6	8	0.9235
60	10.646882	10	6	10	9	6	9	0.9234
61	9.645647	9	7	10	8	7	9	0.9223
62	9.328016	10	6	10	9	6	9	0.9234
63	10.124008	12	6	9	11	6	8	0.9235
64	10.861714	10	6	10	9	6	9	0.9234

65	9.770692	10	6	10	9	6	9	0.9234
66	9.715605	12	7	8	11	7	7	0.9141
67	11.378704	12	6	9	11	6	8	0.9235
68	10.606579	12	6	9	11	6	8	0.9235
69	10.221274	10	6	10	9	6	9	0.9234
70	9.691615	10	6	10	9	6	9	0.9234
71	10.650917	12	6	9	11	6	8	0.9235
72	9.810187	12	6	9	11	6	8	0.9235
73	10.016656	12	6	9	11	6	8	0.9235
74	10.559168	10	9	8	9	9	7	0.9112
75	9.440322	10	7	9	9	7	8	0.9223
76	9.513995	12	6	9	11	6	8	0.9235
77	10.037446	12	6	9	11	6	8	0.9235
78	9.403184	9	7	10	8	7	9	0.9223
79	10.569291	10	6	10	9	6	9	0.9234
80	9.321840	10	6	10	9	6	9	0.9234
81	9.349949	12	6	9	11	6	8	0.9235
82	9.821407	10	6	10	9	6	9	0.9234
83	10.226896	12	6	9	11	6	8	0.9235
84	9.467983	12	7	8	11	7	7	0.9141
85	9.681071	10	7	9	9	7	8	0.9223
86	9.301482	12	6	9	11	6	8	0.9235
87	9.558363	10	7	9	9	7	8	0.9223

88	9.247678	12	6	9	11	6	8	0.9235
89	10.155387	10	6	10	9	6	9	0.9234
90	9.363472	12	6	9	11	6	8	0.9235
91	9.579622	9	7	10	8	7	9	0.9223
92	9.505409	12	6	9	11	6	8	0.9235
93	9.835957	10	6	10	9	6	9	0.9234
94	10.203746	10	7	9	9	7	8	0.9223
95	9.264756	12	6	9	11	6	8	0.9235
96	9.701978	9	8	9	8	8	8	0.9155
97	10.522890	12	7	8	11	7	7	0.9141
98	10.081939	12	6	9	11	6	8	0.9235
99	9.588389	9	8	9	8	8	8	0.9155
100	9.912518	10	7	9	9	7	8	0.9223

۷-٦. نتایج حاصل از ۱۰۰ بار اجرای مثال ۲ توسط الگوریتم رقابت استعماری

	Elapsed time	n 1	n 2	n 3	n 4	r 1	r 2	r 3	r 4	FINAL_AS
1	14/57424	10	5	6	8	9	5	5	6	0/643
2	13/958379	7	6	7	8	6	6	6	6	0/6486
3	14/717952	10	5	6	8	9	5	5	6	0/643
4	13/840782	7	6	7	8	6	6	6	6	0/6486
5	14/109521	8	5	7	8	7	5	6	6	0/6571
6	14/153082	8	5	7	8	7	5	6	6	0/6571
7	14/501753	9	5	8	7	8	5	6	5	0/6312
8	15/037273	10	5	6	8	9	5	5	6	0/643
9	14/885487	8	6	6	8	7	6	5	6	0/6341
10	13/890102	7	7	6	8	6	7	5	6	0/6188
11	13/305652	7	5	6	9	6	5	5	7	0/6342
12	14/063453	9	5	8	7	8	5	7	5	0/6377
13	13/505114	8	4	6	9	7	4	5	7	0/6288
14	13/669536	8	5	7	8	7	5	6	6	0/6571
15	15/059177	9	6	7	7	8	6	6	5	0/6295
16	13/624563	9	4	7	8	8	4	6	6	0/6423
17	15/257023	8	5	7	8	7	5	6	6	0/6571
18	14/088912	8	5	7	8	7	5	6	6	0/6571
19	19/465144	9	6	7	7	8	6	6	5	0/6295
20	13/930123	8	5	7	8	7	5	6	6	0/6571
21	13/43612	8	4	6	9	7	4	5	7	0/6288
22	13/607771	8	5	7	8	7	5	6	6	0/6571
23	14/518735	9	5	8	7	8	5	7	5	0/6377
24	14/478658	8	5	7	8	7	5	6	6	0/6571
25	13/580927	7	5	6	9	6	5	5	7	0/6342
26	14/161799	9	5	8	7	8	5	7	5	0/6377
27	14/115158	9	6	7	7	8	6	6	5	0/6295
28	13/742817	10	5	6	8	9	5	5	6	0/643
29	13/816393	9	5	8	7	8	5	7	5	0/6377
30	14/425536	7	5	6	9	6	5	5	7	0/6342
31	15/401968	10	7	6	7	9	7	5	5	0/6097
32	15/336596	8	7	7	7	7	7	6	5	0/6231
33	13/454962	7	5	6	9	6	5	5	7	0/6342

34	13/551726	7	6	7	8	6	6	6	6	0/6486
35	13/745492	8	5	7	8	7	5	6	6	0/6571
36	13/989913	9	5	8	7	8	5	7	5	0/6377
37	14/272914	10	5	6	8	9	5	5	6	0/643
38	14/869381	8	5	7	8	7	5	6	6	0/6571
39	13/604902	8	5	7	8	7	5	6	6	0/6571
40	15/012205	10	4	8	7	9	4	7	5	0/6177
41	13/977089	7	6	7	8	6	6	6	6	0/6486
42	12/855176	6	4	7	9	5	4	6	7	0/6052
43	13/650074	8	6	6	8	7	6	5	6	0/6341
44	14/000279	9	5	8	7	8	5	7	5	0/6377
45	13/832518	7	6	8	7	6	6	7	5	0/6145
46	13/690909	7	4	8	8	6	4	7	6	0/627
47	14/062756	10	5	6	8	9	5	5	6	0/643
48	13/6977	8	5	7	8	7	5	6	6	0/6571
49	14/504608	7	6	7	8	6	6	6	6	0/6486
50	13/593572	8	6	6	8	7	6	5	6	0/6341
51	13/571025	6	5	8	8	5	5	7	6	0/6183
52	13/616422	7	6	7	8	6	6	6	6	0/6486
53	14/401875	9	6	7	7	8	6	6	5	0/6295
54	14/839441	10	5	6	8	9	5	5	6	0/643
55	15/316818	9	6	7	7	8	6	6	5	0/6295
56	14/53507	8	5	7	8	7	5	6	6	0/6571
57	13/521142	8	5	7	8	7	5	6	6	0/6571
58	14/558067	10	5	6	8	9	5	5	6	0/643
59	13/942113	10	5	6	8	9	5	5	6	0/643
60	13/694068	8	5	7	8	7	5	6	6	0/6571
61	14/103061	8	4	6	9	7	4	5	7	0/6288
62	13/709206	7	6	7	8	6	6	6	6	0/6486
63	13/433672	7	4	8	8	6	4	7	6	0/627
64	14/128332	10	5	6	8	9	5	5	6	0/643
65	15/229708	7	6	7	8	6	6	6	6	0/6486
66	14/069572	10	4	8	7	9	4	7	5	0/6177
67	14/854735	7	6	7	8	6	6	6	6	0/6486
68	13/4674	8	4	6	9	7	4	5	7	0/6288
69	14/288447	8	5	7	8	7	5	6	6	0/6571
70	14/147366	9	5	8	7	8	5	7	5	0/6377
71	13/684464	8	5	7	8	7	5	6	6	0/6571

72	15/341538	10	5	6	8	9	5	5	6	0/643
73	14/515245	6	6	6	9	5	6	5	7	0/612
74	15/294101	8	5	7	8	7	5	6	6	0/6571
75	14/435066	7	8	7	7	6	8	6	5	0/6043
76	14/462828	10	5	6	8	9	5	5	6	0/643
77	13/749554	7	7	6	8	6	7	5	6	0/6188
78	13/797709	9	5	8	7	8	5	7	5	0/6377
79	14/358595	9	5	8	7	8	5	7	5	0/6377
80	13/891341	8	5	7	8	7	5	6	6	0/6571
81	14/237099	10	5	6	8	9	5	5	6	0/643
82	13/303033	8	5	7	8	7	5	6	6	0/6571
83	13/374167	9	4	7	8	8	4	6	6	0/6423
84	13/415089	8	5	7	8	7	5	6	6	0/6571
85	13/137701	8	6	6	8	7	6	5	6	0/6341
86	13/602263	7	6	7	8	6	6	6	6	0/6486
87	13/445248	10	5	6	8	9	5	5	6	0/643
88	13/26409	7	6	7	8	6	6	6	6	0/6486
89	13/610951	7	6	7	8	6	6	6	6	0/6486
90	13/229387	8	5	7	8	7	5	6	6	0/6571
91	13/491817	8	5	7	8	7	5	6	6	0/6571
92	13/560553	7	6	7	8	6	6	6	6	0/6486
93	14/287273	9	5	8	7	8	5	7	5	0/6377
94	14/763862	8	5	7	8	7	5	6	6	0/6571
95	13/744706	7	5	9	7	6	5	8	5	0/6137
96	14/127377	9	6	7	7	8	6	6	5	0/6295
97	14/091162	10	5	6	8	9	5	5	6	0/643
98	14/424111	8	5	7	8	7	5	6	6	0/6571
99	13/586283	7	5	9	7	6	5	8	5	0/6137
100	12/986387	7	4	8	8	6	4	7	6	0/627

Abstract

Optimal design of system and reliability optimization play a key role in engineering design and have been effectively applied to enhance performance.

Failure of products and systems will cause problems at different levels. And can be also considered Even as a serious threat to society and the environment. For this reason in modern society, engineers and technical managers responsible for the planning, design, construction and operation from the simplest product to the most complex system.

One of the most important issues in the reliability and availability of the system, is systems Designing somehow that they have the highest reliability and availability. In literature there are many ways to increase the availability of system components including their reliability, increased levels of redundancy (increase of N), as well as a combination of the two.

In this study, redundancy allocation problem for a system with y subsystem including k-out-of-n structure is investigated. Due to existing repairable components in each subsystem, the model consists of selecting number of repairman and redundancy level for each subsystem. The model is constructed based on Markovian process and the goal is maximization of steady-state availability under constraints such as cost, weight, and volume. In this model, two type of costs are considered; cost of employing repairmen and cost of preparing components and decision variables of the model are number of repairman and number of component in each subsystem.

The model is located into integer non-linear programming with linear constraints. Due to complexity and time-consuming, a heuristic algorithm called Imperialist Competitive is applied to solve the problem. To analyze the model and application of the method, two examples are considered. Finally, to evaluate the method, comparison between the exact solution and solution by the proposed algorithm is done.

Keywords: Availability; Reparability; k-out-of-n system structure; Markovian process; Imperialist Competitive Algorithm.

ISLAMIC AZAD UNIVERSITY

SOUTH THEHRAN BRANCH

FACULTY OF MANAGEMENT AND ACCOUNTING

DEPARTMENT OF INDUSTRIAL MANAGEMEN-OPERATIONS RESEARCH

“M.A.” THESIS

SUBJECT:

Availability Optimization of the k-out-of-n repairable systems using Imperialist Competitive Algorithm

Date:

winter 2015