حل مسئله تخصیص منابع پروژه های چند حالته با منابع محدود بوسیله الگوریتم ژنتیک

مسعود قاسم زاده ۸۲۳۲۰۰۲۵
[masood.ghz@gmail.com]
دانشکده برق ، رایانه و فناوری اطلاعات دانشگاه آزاد واحد قزوین
بهار ۱۳۸۶

چکیده

در این مقاله سعی بر آن شده است تا نوع خاصی از مسائل مرتبط با برنامه ریزی تخصیص منابع در پروژه هایی با چندین حالت اجرایی ، با منابع قابل بازگشت و غیر قابل بازگشت مورد توجه قرار گیرد و برای حل این مسئه از یکی از روش الگوریتم ژنتیک در دو فاز استفاده شده است.هدف از حل این مسئله ارائه یک راه حل و برنامه ریزی تخصیص منابع برای یک پروژه که شامل چندین فعالیت است به نحوی که هر دو محدودیت منابع و شرط پیش نیازی فعالیتها را در پروژه رعایت کرده و این برنامه ریزی منجر به اتمام پروژه در کوتاه ترین زمان ممکن شود.مسائل تخصیص منابع بدلیل گستردگی دامنه جواب آن ، نیاز به کاهش دامنه جواب داشته که سعی شده با استفاده از روش های خاص "حذفی" این دامنه را کوچکتر کرده و سپس با ایجاد جمعیت اولیه تصادفی مرحله اول حل را به جلو برده تا پس از پایان رسیدن مرحله اول به یک مجموعه "برتر" برسیم.این مجموعه "برتر" نود درصد جمعیت اولیه فاز دوم حل مسئله را شامل میشود و ده درصد ما بقی را نیز بصورت تصادفی در نظر گرفته ایم.در نهایت با بررسی این روش با استفاده از ابزار های تولید مسئله الگوریتم مورد تست قرار گرفته است که شاهد پیشرفت قابل ملاحظه ای در رسیدن به جواب بهینه می باشیم.

کلمات کلیدی

تخصیص منابع پروژه – الگوریتم ژنتیک – برنامه ریزی پروژه های چند حالته – برنامه ریزی پروژه منابع محدود.

۱ـ مقدمه

مسئله تخصیص منابع یکی از مسائل قدیمی است که سالیان دراز مورد توجه محققین و دانشمندان بوده است.این گونه مسائل را می توان به انواع مختلفی تقسیم کرد.در یک دید کلی از اینگونه مسائل ، یک پروژه با تعداد n فعالیت نیاز به زمان بندی و تخصیص منابع دارند.این فعالیت ها می بایست طبق محدودیت های از پیش تعریف شده زمانبندی شوند.بعنوان مثال شروع شدن یک فعالیت قبل از اتمام یک و یا چند فعالیت قبلی

مجاز نمی باشد.همچنین با توجه به نوع مسئله منابع نیز میتوانند دارای محدودیت باشند.در این مقاله به نوع خاصی از مسائل تخصیص منابع با منابع محدود (RCPSP) پرداخته شده است که علاوه بر محدود بودن منابع ، منابع مورد استفاده در دو دسته "قابل استفاده مجدد" و "غیر قابل استفاده مجدد" دسته بندی شده اند.همچنین هر یک ار فعالیت های مورد نظر می توانند در چند حالت مختلف اجرا شده که بر خلاف مسائل تک حالته (RCPSP) ، بسته به حالت اجرایی آنها ، مدت زمان اجرا و منابع مورد نیاز نیز متفاوت خواهند بود.این نوع خاص از مسئله تخصیص منابع که به "مسئله تخصیص منابع پروژه های چند حالته با منابع محدود" (MRCRSP) معروف است جزو کاملترین مسائل در بحث تخصیص منابع به شمار میرود.هدف از حل این مسئله ، زمانبندی و تخصیص فعالیت های یک پروژه با توجه به محدودیت های پیش نیاز و پس نیازی و همچنین محدودیت های منابع ، جهت رسیدن به بهترین "طول زمان پروژه" (makespan) است.
برای نشان دادن میزان پیچیدگی مسائل و راه حل های ارائه شده برای آنها ، علم تئوری پیچیدگی ، مسائل را به دسته های مختلف easy و hard تقسیم میشوند.(Garey and Johnson 1979 , Karp 1975 Lenstra and Rinnooy Kan 1979 , Shmoys 1993 , Stockmeyer 1992) که این دسته بندی ارتباط مستقیمی با الگوریت ارائه شده و پیچیدگی زمانی آن الگوریت دارد.مسائلی که الگوریتم های ارائه شده برای آنها دارای پیچیدگی زمانی خطی باشند ، easy ، و مسائلی که پیچیدگی زمانی الگوریم ارائه شده برای آن غیرخطی و صعودی است ، از نوع hard تلقی میشوند.مسائل hard نیز بسته به نوع مسئله و الگوریتم حل آن به انواع مختلفی تقسیم میشود.با توجه به اینکه مسئله تخصیص منابع محدود(RCPSP) یک حالت کلی ازمسائل static job shopping است در نتیجه این مسئله (RCPSP) و مسئله تخصیص منابع محدود چند حالته (MRCPSP) نیز جزو دسته مسائل NP-har محسوب می شود(Blazewicz et al. 1983).
دانشمندان زیادی بر روی این مسئله کار کرده و روش های مختلف heuristic و metaheuristic جهت حل آن بکار برده اند.
Slowski et al.[18] بر روی رهیافت تک مسیره ، چند مسیره ، و الگوریتم Simulated annealing کار کرده اند. Kolisch and Drexl[13] با استفاده از فرموله کردن مسئله MRCPSP به صورت "برنامه ریزی صفر و یک" و ارائه یک الگوریتم جستجوی محلی ، اهمیت آن را با پیاده سازی یک برنامه کاربردی برای مدیریت تولید و عملیات نشان داده اند. Boctor[3][4][5] یک الگوریتم heuristic برای حل این مسئله پیشنهاد کرده است که در آن منابع خیر قابل استفاده مجدد مد نظر قرار نگرفته است.طبق گفته Hartman and Drexl[10] بهترین الگوریتمی که برای این مسئله ارائه شده است توسط Sprecher and Drexl[19] بوده که در آن از روش "شاخه و برگ" استفاده کرده اند.
همچنین روش های metaheuristic نیز برای حل این مسئله مورد استفاده قرار گرفته اند.از جمله Bouleimen and Lecocp[6] and Jozefowska et al.[12] بر روی روش های "simulated annealing"کار کرده اند. Bouleimen and Lecocq[6] از دو حلقه جستجو برای حل مسئله استفاده کرده اند. Jozefowska et al.[12] از روش neighborhood برای حل مسئله بصورت تغییر فعالیت ، تغییر حالت اجرا و یا هر دو استفاده کرده است.اما افرادی نیز برای حل مسئله از روش مد نظر در این مقاله یعنی الگوریتم ژنتیک استفاده کرده اند که به تعدادی از آنها اشاره خواهد شد. Hartman[8] and Alcaraz et al.[1] یک روش مبتنی بر الگوریتم ژنتیک برای حل مسئله RCPSP ارائه داده اند.همچنین آنها این روش را برای حل مسائل MRCPSP نیز گسترش داده اند.در مسئله ای که Tseng[16] and Ozdamar[17] برای حل استفاده کرده اند ، فقط منابع قابل استفاده مجدد مد نظر قرار گرفته شده است.البته افراد دیگری نیز بر روی این مسئله کار کرده اند که هر یک به نوعی از روش های مختلف برای حل مسئله استفاده کرده اند.
همانطور که قبلا اشاره شد ، مسئله ای که در حال حاضر بر روی آن کار شده است ، از نوع کامل این مسئله بوده و با در نظر گرفتن محدودیت منابع و همچنین تقسیم منابع به دو دسته قابل استفاده مجدد و غیر قابل استفاده مجدد ، سعی دارد تا این محدودیتها را در قالب یک مسئله چند حالته حل کند که برای حل آن از الگوریتم های ژنتیک در دو مرحله استفاده میکند.

۲ـ شرح مسئله

مسئله مورد نظر در یک دید کلی یک پروژه است که از $J=\{1,2,3\ldots n\}$ فعالیت تشکیل شده است و طول دوره هر فعالیت با $D(j)$ بیان میشود.هر یک از این فعالیت ها می توانند در $EM(j)=\{1,2,3\ldots M(j)\}$ حالت مختلف اجرا شوند.تفاوت بین این حالتهای مختلف در ادامه بیان خواهند شد.در این مسئله دو نوع منبع وجود دارد.مجموعه منابع "قابل استفاده مجدد" که با RR بیان میشوند و مجموعه منابع "غیرقابل استفاده مجدد" NR . همانطور که از اسم این منابع نیز مشخص است ، منابعی که "قابل استفاده مجدد" هستند پس از

اینکه یک بار مورد استفاده قرار گرفتند ، پس از اینکه کار با آن منبع تمام شد ، مجددا قابل استفاده در یک فعالیت دیگر می باشند.بعنوان نمونه هنگامی که یک دستگاه دستگاه رایانه شخصی در یکی از مراحل انجام پروژه مورد استفاده قرار میگیرد ، پس از اتمام آن مرحله میتوان از آن رایانه در مراحل بعدی نیز استفاده مجدد کرپس رایانه را میتوان یک منبع "قابل استفاده مجدد" دانست.اما در نظر بگیرید که در یکی از مراحل پیاده سازی زیر ساخت های شبکه در یک سازمان ، نیاز به استفاده از کابل های انتقل اطلاعات برای ایجاد بستر مناسب شبکه است.اگر از این منبع – کابل انتقال اطلاعات- در یکی از مراحل استفاده شود ، در مراحل بعدی امکان استفاده مجدد از آن منبع وجود ندارد چرا که آن منبع مصرف شده است و غیر قابل استفاده مجدد می باشد.

در بیان این مسئله $m(j)$ نشان دهنده حالت اجرایی فعالیت $j \in J$ ، و $d(mj)$ بیان کننده طول دوره اجرای فعالیت j در حالت اجرایی m و در نهایت $r_{jm,k}^{RR}$ بیانگر تعداد واحد منبع قابل استفاده مجدد واحد $k \in RR$ و $r_{jm,k}^{NR}$ بیانگر تعداد واحد منبع غیرقابل استفاده مجدد $k \in NR$ است.

تعداد واحدهای منابع "قابل استفاده مجدد" و "غیرقابل استفاده مجدد" بترتیب برابر با تعداد واحد ثابت Q_k^{RR} و Q_k^{NR} میباشد به قسمی که $k \in RR$ و $k \in NR$ است.در نظر میگیریم تمامی فعالیتهایی در در لحظه t در حال انجام هستند را با $A(t)$ و $m(j) \in EM(j)$ بیانگر حالت اجرایی آن فعالیتها باشند.با توجه به این تعاریف محدودیت منابع "قابل استفاده مجدد" را به صورت زیر تعریف میکنیم :

$$\sum_{j \in A(t)} r_{jm,k}^{RR} \le Q_K^{RR}, \forall k \in RR \ and \ \forall t$$

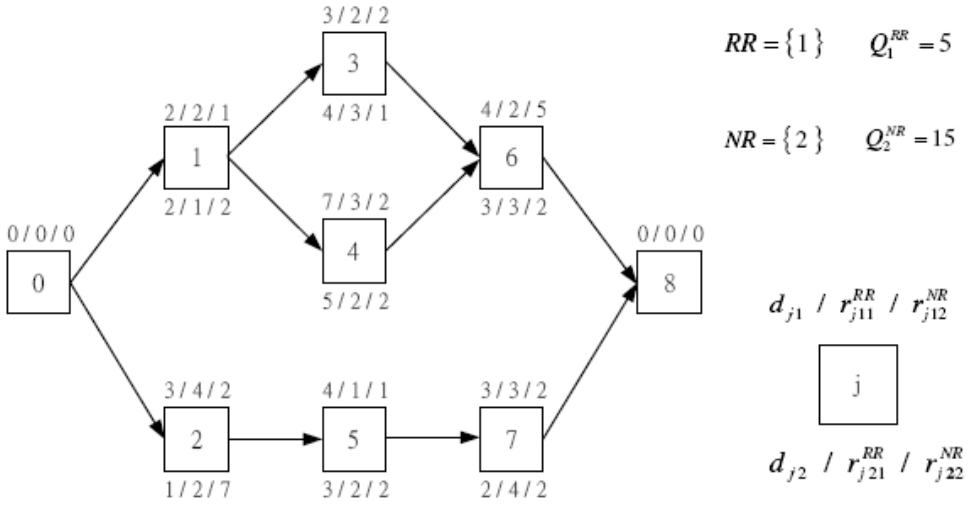
و در منابع "غیر قابل استفاده مجدد" داریم :

$$\sum_{j \in A(t)} r_{jm,k}^{NR} \le Q_K^{NR}, \forall k \in NR$$

در نظر میگیریم $P(j)$ مجموعه فعالیتهای پیش نیاز فعالیت $j \in J$ باشد.پیش نیاز بودن بدین معناست که فعالیت j نمیتواند زودتر از اتمام تمام فعالیتهای پیش نیاز خود که با مجموعه $P(j)$ مشخص میشوند شروع شود.طبیعتا از خصوصیات و ویژگی های هر مسئله تخصیص منابع و کنترل پروژه داشتن این چنین محدودیت هایی است.با توجه به اینکه برای نشان دادن مسئله از روش نمایش شبکه "فعالیت بر نود" استفاده شده است ، وجود دو فعالیت غیر حقیقی 0 و $n+1$ بر روی شبکه بیانگر این امر است که این دو فعالیت صرفا جهت نمایش شروع و پایان پروژه است و هیچگونه منبعی را مصرف نمیکنند.

هدف از حل این مسئله یافتن یک برنامه تخصیص منابع در کوتاه ترین زمان که تمامی محدودیت های ما از جمله : پیش نیازی ها ، محدودیت منابع قابل استفاده مجدد و محدودیت منابع غیر قابل استفاده مجدد را ارضاء کند.طبیعی است که تمامی فرضیات در مورد منابع بصورت یک عدد مثبت در نظر گرفته شده است.

در شکل زیر یک نمونه از مسئله ارائه شده است که سعی میشود تا پایان همین مسئله مورد توجه قرار گیرد.



شکل ۱ – نمونه ای از یک پروژه

۳- الگوریتم ژنتیک دو فاز

در الگوریتمی که برای حل این مسئله ارائه شده است ، سعی بر آن شده است تا مسئله در دو مرحله حل شود.در مرحله اول به نوعی یک جستجوی کلی بر روی دامنه جوابهای مسئله انجام شده و در مرحله دوم به صورت دقیق تر تمرکز الگوریتم بر روی مناطق با قابلیت بهتر و نزدیکتر به جواب و بصورت جزئی تر نگاه میشود.با توجه به این نگرش ، جمعیت اولیه تشکیل دهنده فاز اولیه ، بصورت تصادفی انتخاب شده و در مرحله دوم اکثر جمعیت اولیه را نتایج حاصل از فاز اول مسئله تشکیل میدهند.در ادامه سعی میشود تا توضیحات دقیق تری پیرامون روش حل این مسئله ارائه شود.

۳- ۱ نمایش کروموزم و تابع هدف

همانطور که قبلا نیز اشاره شد بع علت اینکه پروژه های با توانایی اجرا شده به صورت چند حالته مد نظر است فلذا به هنگام حل و ارائه کروموزم نیز میبایست به نوعی "حالت" اجرایی هر فعالیت را نیز مورد توجه قرار داد.به همین منظور کروموزم به صورت یک بردار دو سطری $I = (\lambda, \mu)^T$ نمایش داده میشود که سطر اول آن بیانگر فعالیت و سطر دوم آن "حالت" اجرایی آن فعالیت را نشان میدهد.بعنوان مثال بردار زیر بیانگر یک کروموزوم است که فعالیتها و حالت های اجرایی آنرا نشان میدهد.این کروموزوم میتواند یک راه حل برای نمونه ارائه شده در شکل ۱ میباشد.

$$I(1) = \begin{pmatrix} 0 & 1 & 2 & 4 & 5 & 7 & 3 & 6 & 8 \\ 1 & 2 & 1 & 2 & 2 & 2 & 1 & 2 & 1 \end{pmatrix}$$

تابع هدف در این روش ، محاسبه "زمان" (makespan) آن کروموزم است.روش های مختلفی برای محاسبه زمان یک پروژه وجود دارد که یکی از کاربردی ترین روش ها استفاده از روش تخصیص"جلوگرد" (forward Scheduling) و روش تخصیص "عقب گرد" (Backward Scheduling) است که روش حل این مسئله بسته به نوع آن متفاوت است.در روش "جلوگرد" سعی بر آن میشود تا هر فعالیت با توجه به محدودیتهای خود در زودترین زمان ممکن شروع شود.بدین معنی که یک فعالیت که شرایط پیش فرضی آن رعایت شده یعنی اتمام فعالیت های قبلی و پیش نیاز آن شروع شده است سعی میکند تا در زودترین زمان ممکن شروع شود.اگر که محدودیتهای منابع اجازه این کار را ندهند یعنی برای انجام و شروع فعالیت مد نظر با کمبود منبع مواجه شویم ، فعالیت مذکور آنقدر به تاخیر می افتد تا منابع مورد نیاز آزاد شده (فعالیتهای دیگری که منابع را در اختیار گرفته اند به اتمام برسند) و سپس این فعالیت در زودتریت زمان ممکنه فعالیت

خود را آغاز میکند.به طور مشابه روش "عقب گرد" سعی بر آن دارد تا از یک زمان فرضی اتمام پروژه شروع کرده و فعالیتها را در دیرترین زمان ممکن و با توجه به محدودیتها برنامه ریزی میکند.
"زمان" حاصل شده که به روش ذکر شده بدست می آید بعنوان "تابع هدف" (Fitness function) در نظر گرفته میشود.البته باید توجه داشت که روش "جلوگرد" و یا "عقب گرد" تمرکز بر روی برنامه ریزی مسائل با منابع محدود قابل استفاده مجدد هدارند.این بدین معناست که به هنگام برنامه ریزی یک مسئله ، روش "عقب گرد" یا "جلوگرد" سعی میکند تا مسئله را با محدودیت "منابع قابل استفاده مجدد" حل کرده و توجهی به محدودیت "منابع غیر قابل استفاده مجدد" ندارد.پس این انتظار میرود که پس از محاسبه "زمان" پروژه توسط یکی از روش های "جلوگرد" و یا "عقب گرد" ، شاهد برنامه ریزی هایی – کروموزم ها - باشیم که محدودیت "منابع غیر قابل استفاده مجدد" را ارضاء نکرده اند.به همین منظور برای محاسبه "تابع هدف" از روشی که در [2] ارائه شده است برای تعیین این دو دسته کروموزم استفاده شده است."تابع هدف" بصورت زیر محاسبه میشود.

$$f(I)=\begin{cases}1)mak(I) & \text{If } I \text{ is feasible}\\ 2)\max\_\ fea\_pop\_mak+mak(I) & \text{If } I \text{ is infeasible}\\ -\min\_\ project\_CC+SFT(I)\end{cases} \qquad (3)$$

در تعریفی که در (3) ارائه شد و روشی که برای محاسبه "زمان" پروژه محاسبه مشود بعلت اینکه ایم مقدار کوچک است میتواند بعنوان یک "تابع هدف" خوب در نظر گرفته شود
در تعریف (3) داریم ، make(I) برابر است با "زمان" پروژه که با یکی از روش های "جلوگرد" و یا "عقب گرد" محاسبه میشود.عبارت max_fea_pop_mak برابر با بیشترین مقدار "زمان" پروژه در نسل مورد نظر است.همچنین min_project_CC برابر با "مسیر حیاتی" آن پروژه است."مسیر حیاتی" به طول کوتاه ترین مسیر یک پروژه ابلاغ میشود که برار با حاصل جمع "زمان"اجرای هر فعالیت در "مسیر حیاتی" است.در واقع "مسیر حیاتی" شامل یک مجموعه از فعالیت هاست که هرگونه تغییر زمانی فعالیتهای این مجموعه منجر به کوتاه شدن و یا بلند شدن "زمان" پروژه میشود حال آنکه ممکن است جابجایی و یا تغییر زمان اجرای دیگر فعالیتها – فعالیتهایی که در مجموعه"مسیر حیاتی" نیستند- در "زمان" کل پروژه تغییری بوجود نیاورند.و در نهایت SFT(I) نمایانگر تعداد واحد "منبع غیر قابل استفاده مجدد"ی است که در کروموزم مذکور از سقف محدودیت منبع تجاوز کرده اند.در ادامه سعی میشود تا یک مثال برای روشن شدن مطلب ارائه شود.فرض میکنیم نمونه پروزه ای که در شکل ۱ ارائه شد پس از برنامه ریزی توسط یکی از روش های "جلوگرد" و یا "عقب گرد" به شکل زیر ارائه شده است :

$$I(1)=\begin{pmatrix}0&1&2&4&5&7&3&6&8\\1&2&1&2&2&2&1&2&1\end{pmatrix}$$

$$I(2)=\begin{pmatrix}0&1&2&3&4&5&6&7&8\\1&2&2&1&2&2&1&2&1\end{pmatrix}$$

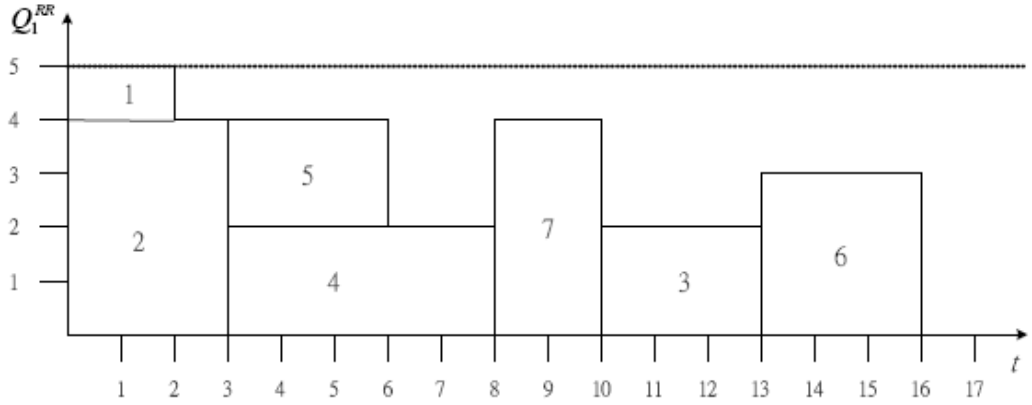
همانطور که مشاهده میشود بدلیل اینکه I(1) تعداد ۱۴ واحد از "منابع غیر قابل استفاده مجدد" را نیاز دارد، پس I(1) ارضاء شده است. این تعداد واحد از تعداد کل "منابع غیر قابل استفاده مجدد" که ۱۵ است کمتر می باشد.اما در مورد I(2) وضع متفاوت است.این کروموزم نیاز به ۲۲ واحد "منبع غیر قابل استفاده مجدد دارد که از مقدار منبع موجود بیشتر است و به نوعی از سقف این محدودیت تجاوز میکند در نتیجه کروموزم I(2) یک کروموزم ارضاء نشده است

.با توجه به آنچه که در (3) ارائه شد داریم :

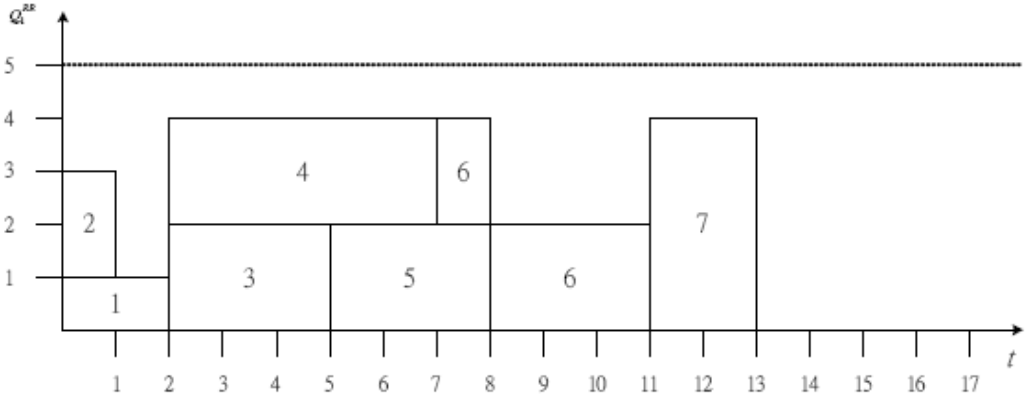
I) mak(I(1)) = 16 and I(1) is feasible => f({I1)) = 16

II) mak(I(2))=13 and I(2) is infeasible =>
Max_fea_pop_mak=19 and min_project_CC=10 and SFT(I(2))=7
F(I(2)) = 29

در شکل های زیر راه حل های I(1) و I(2) نمایش داده شده اند.



برنامه ریزی "جلوگرد" برای کروموزم I(1)



برنامه ریزی "جلوگرد" برای کروموزم I(2)

لازم بذکر است که برای محاسبه "زمان" پروژه از روش SGS Serial[Kelley 1963] استفاده شده است.

۴ ـ الگوریتم ژنتیک

الگوریت ژنتیک یکی از روش های بکار گرفته شده برای حل مسائلی NP-hard می باشد که با الگو برداری از طبیعت سعی بر آن دارد تا مسائل را بصورت یک مدل منطبق بر ساختار ژن در آورده و آن را حل نماید.

۴ـ۱ جمعیت اولیه و مجموعه برتر

در فاز اولیه این روش حل ، جمعیت اولیه بصورت تصادفی تولید میشوند بدین ترتیب که هر کروموزم شامل دسته فعالیتها و حالتهای مربوطه به آن فعالیت است که بصورت تصادفی این کروموزم ایجاد میشود.شایان ذکر است که تنها محدودیتی که به هنگام تعریف جمعیت اولیه فاز نخست اعمال میشود ، شرط رعایت پیش نیازی در کروموزم ایجاد شده است.بدین معنا که در صورتی که در جمعیت اولیه کروموزمی مشاهده شد که پیش نیازی فعالیتها را رعایت نکرده بود ، با یک کروموزم دیگر که شرط پیش نیازی فعالیتها را رعایت

میکند تعویض می شود.از آنجایی که ما مطمئن هستیم که استفاده از روش های "جلوگرد" و یا "عقب گرد" محدودیتهای مرتبط با "منابع قابل استفاده مجدد" را ارضاء میکند پس تنها حالتی که ممکن است برای یک کروموزم بوجود بیاید عدم رعایت و ارضاء محدودیت "منابع غیر قابل استفاده مجدد" است.برای جلوگیری از این کار به هنگام ایجاد جمعیت اولیه فاز نخست ، کروموزم هایی که از محدوده "منابع غیر قابل استفاده مجدد" تجاوز کرده اند انتخاب شده و یک "عامل" سعی در تغییر این کروموزم به یک کروموزم ارضاء شده می نماید بدین ترتیب که به صورت تصادفی یکی از فعالیتها انتخاب شده و "حالت" ابلاغ شده به آن فعالیت به یک "حالت" با تعداد "منابع غیر قابل استفاده مجدد" کمتر تبدیل میگردد.این عمل آنقدر ادامه پیدا میکند تا کروموزم مورد نظر هر دو محدودیت "پیش نیازی" و "منابع غیر قابل استفاده مجدد" را ارضاء کند.پس از پایان فاز اول این الگوریتم ، مجموعه "برتر" حاصل میشد که این مجموعه شامل کروموزم هایی با تابع هدف بهتر هستند.از این مجموعه "برتر" برای ایجاد جمعیت اولیه فاز دوم استفاده می شود.در واقع ۹۰ درصد جمعیت اولیه فاز دوم راه حل را ، مجموعه "برتر" فاز اول تشکیل می دهد و ۱۰ درصد مابقی بصورت تصادفی ایجاد میشود.

corssover ۲ – ۴

برای حل این مسئله ، از دو الگوریتم crossover بر پایه two-point crossover استفاده شده است.که در ادامه سعی میشود تا این دو روش را توضیح دهیم.
در crossover(1) ، دو نقطه برش وجود دارد بنام های cp(1) و cp(2) که cp(1)<cp(2) است.فرض کنید دو کروموزم $I^F = (\lambda^F, \mu^F)^T$ و $I^M = (\lambda^M, \mu^M)^T$ بترتیب بعنوان کروموزم های "پدر" و "مادر" انتخاب شده اند.پس از اعمال crossover بر روی این دوکروموزم ، دو کروموزم فرزند بصورت $I^S = (\lambda^S, \mu^S)^T$ و $I^D = (\lambda^D, \mu^D)^T$ بوجود می آیند که طریقه تشکیل این دو فرزند بدین صورت است :
دو نقطه برش بر روی کروموزم ها بصورت تصادفی مشخص میشوند.فرزند I(s) قسمت اول ژن های خود را از تکه اول کروموزم I(f) به ارث میبرد.تکه دوم-میانی- کروموزم I(s) از قطعه cp(2)-cp(1) کروموزم I(m) -یعنی قطعه وسط- حاصل میشود و در نهایت باقیمانده کروموزم I(s) را قطعه cp(2) کروموزم I(f) - همان قطع انتهایی- تشکیل میدهد.در مورد کروموزم I(d) نیز وضعیت مشابه به آنچه که در مورد کروموزم I(s) بیان شد است با این تفاوت که قسمت اول کروموزم I(d) از قطعه اول I(m) و قطعه دوم از قطعه وسط I(f) و بهین ترتیب ادامه داده میشود.تمامی "حالت" هایی که در فرزندان ایجاد میشود دقیقا برابر با "حالت" های والدینشان است و هیچگونه تغییری در "حالت" اجرایی آنها بوجود نمی آید.
در crossover(2) نیز روش تقسیم و اختصاص فعالیتها در فرزندان کاملا مشابه با روش crossover(1) است با این تفاوت که "حالت" هایی که به فعالیتها ی فرزندان تخصیص داده میشود مشابه با "حالت" هایی اجرایی والدین آنها نیست.در crossover(2) برای اختصاص "حالت" اجرایی ، یک رشته اعداد تصادفی 0و 1 به فعالیتها اختصاص داده میشود.اگر این عدد تصادفی برابر با 1بود ، $\lambda^D(I) = \lambda^M$ و $\lambda^S(I) = \lambda^F$ و در صورتی که عدد منتصب شده برابر با 0 بود داریم : $\lambda^D(I) = \lambda^F$ و $\lambda^S(I) = \lambda^M$.

Mutation ۵ –۴

برای mutation نیز دو روش به کار رفته است.در mutation(1) هر کروموزم به دو مجموعه "الف" و "ب" تقسیم میشود که این تقسیم بندی بر اساس یک احتمال از پیش تعیین شده است و ممکن است اعضای دو مجموعه باهم اشتراک داشته باشند.پس از این تقسیم بندی یک تابع سعی میکند تا جایگاه فعالیت ها عوض شده و به جای دیگری در همان مجموعه منتقل شود و یک ترکیب جدید بوجود آید که محدودیت "پیش نیازی" را نیز ارضاء می کند."حالت" های این مجموعه عوض نشده و حالت قبلی خود را دارند.در مجموعه "ب" تنها سعی میشود تا "حالت" اجرایی هر فعالیت بصورت تصادفی عوض شود.
در mutation(2) ما سعی میکنیم تا مجموعه فعالیت های "مسیر بحرانی" را به دو دسته "الف" و "ب" تقسیم کرده و دقیقا همان فعالیتی که بر روی مجموعه های "الف" و "ب" mutation(1) انجام می دادیم را بر روی این دو مجموعه نیز اجرا کنیم.

۶ –۴ انتخاب (selection)

در مرحله "انتخاب" سعی شده است تا از دو روش متفاوت بنام های "rank selection" و "2-tournoment" استفاده شده است.در این دو روش بترتیب در مرحله اول و دوم استفاده شده است.

۵ ـ الگوریتم ژنتیک

همانطور که بیان شد این روش در دو مرحله به حل مسئله پرداخته است.در ابتدا جمعیت اولیه بصورت کاملا تصادفی ایجاد شده و پس از گذر از crossover(1) و crossover(2) به قدم بعدی mutation رفته و در نهایت با شرط خاتمه مرحله اول ، یک جمعیت "نخبه" بوجود می آیند که این جمعیت نود درصد جمعیت اولیه مرحله دوم را تشکیل می دهند.ده درصد ما بقی جمعیت نیز بصورت تصادفی ایجاد میشوند.در شکل ۳ شبه کد این الگوریتم مشاهده میشود.

While (termination condition not met)
Generate initial population
While (*loop* < *loopstuck*)
While (*generation* < *genstuck*)
Apply crossover operator
Apply Mut_1 operator
Evaluate fitness and apply forward-backward local search operator
Update Elite Set
Ranking selection
If (best solution is improved) *generation* = 0
Else *generation*++
End while
Apply Mut_2 operator
Evaluate fitness and apply forward-backward local search operator
Update Elite Set
2-tournament selection
loop++
End while
End while

شکل ۳ – شبه کد الگوریتم ژنتیک

۶ ـ نتیجه گیری

با توجه به آنکه پروژه در حال انجام است و پیاده سازی آن بطور کامل به اتمام نرسیده است و نتیجه قطعی آن قابل اثبات کامل نیست اما با توجه به آنچه که در جدول زیر قابل مشاهده است می توان انتظار داشت با روش های بکار گرفته شده در جهت بهبود این روش حل ، نتیجه کار بهینه تر باشد.همچنین قابل یاد آروی است که با توجه به اینکه کار بطور کامل به اتمام نرسیده است ، امکان ایجاد تغییراتی در روش حل مسئله از جمله روش های بکار رفته در crossover و یا mutation وجود دارد.برای اندازه گیری و مقایسه این روش استفاده از ابزارهای تست مختلفی وجود دارد که در این مقاله ما سعی می کنیم تا با استفاده از یک ابزار تولید مسئله که توسط Prof.Kolisch ارائه شده است و تحت عنوان PSPLIB معروف است استفاده کنیم.
PSPLIB یکی از ابزارهای تولید مسئله است که دارای بسته های مختلف مسئله می باشد.این بسته ها بسته به نوع آنها دارای تعداد مختلفی پروژه که هر یک از پروژه ها دارای تعداد خاصی فعالیت جهت تست و اعمال برنامه ریزی بر روی آنهاست.به عنوان مثال بسته J300 این ابزار تولید مسئله دارای 640 پروژه در خود است.در نهایت سعی بر آن شده است تا حالت تکمیلی این مسئله مورد توجه قرار گیرد.

مراجع

[1] J. Alcaraz and C. Maroto, “A robust genetic algorithm for resource allocation in

project scheduling", *Ann. Oper. Res.*, vol. 102, pp.83-109, 2001.
[2] J. Alcaraz, C. Maroto and R. Ruiz, "Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms", *J. Oper. Res. Soc.*, vol. 54, pp. 614-626, 2003.
27
[3] F.F. Boctor, "Heuristics for scheduling projects with resource restrictions and several resource-duration modes", *Int. J. Prod. Res.*, vol. 31, pp. 2547-2558, 1993.
[4] F.F. Boctor, "An adaption of the simulated annealing algorithm for solving the resource-constrained project scheduling problems", *Int. J. Prod. Res.*, vol. 34, pp. 2335-2351, 1996.
[5] F.F. Boctor, "A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes", *Eur. J. Oper. Res.*, vol. 90, pp. 349-361, 1996.
[6] K. Bouleimen and H. Lecocq, "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version", *Eur. J. Oper. Res.*, vol. 149, pp. 268-281, 2003.
[7] A. Drexl and J. Grünewald, "Nonpreemptive multi-mode resource-constrained project scheduling", *IIE Trans.*, vol. 25, pp. 74-81, 1993.
[8] S. Hartmann, "A competitive genetic algorithm for resource-constrained project scheduling", *Nav. Res. Logist.*, vol. 45, pp. 733-750, 1998.
[9] S. Hartmann, "Project scheduling with multiple modes: A genetic algorithm", *Ann.*
Oper. Res., vol. 102, pp. 111-135, 2001.
[10] S. Hartmann and A. Drexl, "Project scheduling with multiple modes: A comparison of exact algorithm", *Networks*, vol. 32, pp. 733-750, 1998.
28
[11] S. Hartmann and R. Kolisch, "Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem", *Eur. J. Oper. Res.*, vol. 127, pp. 394-407, 2000.
[12] J. Jozefowska, M. Mika, R. Rozychi, G. Waligora and J. Weglarz, "Simulated annealing for multi-mode resource-constrained project scheduling", *Ann. Oper. Res.*, vol. 102, pp. 137-155, 2001.
[13] R. Kolisch and A. Drexl, "Local search for nonpreemptive multi-mode resource-constrained project scheduling", *IIE Trans.*, vol. 43, pp. 987-999, 1996.
[14] R. Kolisch and S. Hartmann, "Experimental investigation of heuristics for resource-constrained project scheduling: an update", working paper, Technical University of Munich, Germany, 2004. Available: http://halfrunt.bwl.uni-kiel.de/bwlinstitute/Prod/mab/hartmann/hartmann.html .
[15] R. Kolisch and A. Sprecher, "PSPLIB – a project scheduling problem library", *Eur. J. Oper. Res.*, vol. 96, pp. 205-216, 1996.
[16] M. Mori and C.C. Tseng, "A genetic algorithm for multi-mode resource constrained project scheduling problem", *Eur. J. Oper. Res.*, vol. 100, pp. 134-141, 1997.
[17] L. Özdamar, "A genetic algorithm approach to a general category project scheduling problem", *IEEE Trans. Sys. Man, Cybern. Part C.*, vol. 29, pp. 44-59,
29
1999.
[18] R. Slowinski, B. Soniewicki and J. W_glarz, "DSS for multiobjective project scheduling subject to multiple-category resource constraints", *Eur. J. Oper. Res.*, vol. 79, pp. 220-229, 1994.
[19] A. Sprecher and A. Drexl, "Multi-mode resource-constrained project scheduling

by a simple, general and powerful sequencing algorithm", *Eur. J. Oper. Res.*, vol. 107, pp. 431-450, 1998.
[20] A. Sprecher, S. Hartmann and A. Drexl, "An exact algorithm for project scheduling with multiple modes", *OR Spectrum*, vol. 19, pp. 195-203, 1997.
[21] L. Y. Tseng and S. C. Chen, "A hybrid metaheuristic ANGEL for the resource-constrained project scheduling problem", To appear on *Eur. J. Oper. Res.*