International Journal of Industrial Engineering & Production Management (2017)

September 2017, Volume 28, Number 3
Pp 375- 390

http://IJIEPM.iust.ac.ir/

Tuning Parameters for SAT Solver Problem in Multi-Mode Resource Constrained Project Scheduling Problem

Razeyeh Cheshomi, Hamed Reza Taregian*& Hamid Reza Yoosefzadeh

Razeyeh Cheshomi, *Master of Science, Faculty of Mathematical Sciences, FerdowsiUniversity of Mashhad*
Hamid Reza Yoosefzadeh, *Assistant Professor, Department of Mathematical Sciences, Payame Noor University*
Hamed Reza Taregian, *Full Professor, Faculty of Mathematical Sciences, FerdowsiUniversity of Mashhad*

Keywords

Project scheduling,
SAT solver,
Multi-mode RCPSP,
Enumeration.

ABSTRACT

Multi-mode resource constrained project scheduling problem (MRCPSP) is a generalization of the resource constrained project scheduling problem (RCPSP). The objective of the MRCPSP is to determine a schedule with minimum makespan by selecting exactly one mode of execution for each activity subject to the precedence constraints as well as the renewable and non-renewable resource constraints. New algorithm splits the MRCPSP into two steps. First step, borrowing ideas from the Boolean algebra, enumeration schemes and satisfiability (SAT) problem solver the problem is converted into a single mode project scheduling problem(RCPSP) and twice step, is then solved using an efficient meta-heuristic procedure from the literature. The main problem with this method is, it often requires a higher CPU times and also larger memory. In order to improve the performance of the method, in this article propose three rules: ordering and two rule of pruning. Our experimental results show that in majority of cases, implementation of our rules lead to significant improvements regarding computer running time and the need for memory about 88 percent.

نشـــریه بین المللـــی مهنـــدسی صنـــایع و مدیـــــریت تولیـــد

شمـــاره ۳، جلـــد ۲۸، پاییز۱۳۹۶
صفحـــه ۳۷۶- ۳۹۰

ISSN: 2008-4870

http://IJIEPM.iust.ac.ir/

بهسازی پارامترهای حل‌کننده مساله صدق‌پذیری برای زمان‌بندی پروژه با منابع محدود چندحالته

راضیه چشمی، حامدرضا طارقیان * و حمیدرضا یوسف زاده

کلمات کلیدی

زمان‌بندی پروژه، حل‌کننده مساله صدق‌پذیری، درخت شمارشی، مساله زمان‌بندی پروژه با منابع محدود چندحالته.

چکیده:

مساله زمان‌بندی پروژه با منابع محدود چندحالته تعمیم مساله زمان‌بندی پروژه با منابع محدود تک‌حالته است. هدف آن، انتخاب یک حالت برای اجرای هر فعالیت است تا پروژه را با توجه به محدودیت‌های پیش‌نیازی، منابع تجدیدپذیر و منابع تجدیدناپذیر در زمان کمینه زمان‌بندی کند. در روشی که اخیرا برای حل آن ارایه شده، این مساله در دو گام مجزا حل می‌شود. در گام نخست، با استفاده از مفاهیم جبر بول، درخت شمارشی وهمچنین حل‌کننده مساله صدق‌پذیری، مساله زمان‌بندی پروژه با منابع محدود چندحالته به یک مساله زمان‌بندی پروژه با منابع محدود تک حالته تبدیل شده ودر گام دوم این مساله تک حالته با کمک یکی از الگوریتم‌های زمان‌بندی حل می‌شود. مشکل اساسی این روش، کم آوردن حافظه و زمان اجرای طولانی آن است. در این مقاله و برای رفع مشکلات یاد شده، سه راهکارجدید یعنی مرتب‌سازی و دو قاعده هرس درخت شمارشی پیشنهاد می‌شود و اثرات آنها مورد آزمون قرار می‌گیرد. نتایج آزمون‌ها نشان می‌دهند که به کارگیری پیشنهادات در تعدیل مشکلات روش حل به میزان قابل توجهی موثر بوده و براساس معیارهای تعیین شده توانسته است در مواردی بیش از ۸۸ درصد روند را بهبود بخشد. از این طریق نه تنها سرعت پردازش ارتقا یافته بلکه در میزان حافظه مورد نیاز نیز صرفه‌جویی شده است.

۱. مقدمه

در دنیای رقابتی امروز که با سرعت فزاینده ای در حال پیشرفت است، بهره برداری بهینه از زمان و سرمایه نقش بسزایی در حیات اثر بخش سازمان ها ایفا می کند. از این رو، علومی هم که به نوعی بهینه سازی این عوامل را مورد توجه قرار می‌دهند، اهمیت ویژه‌ای می‌یابند. مدیریت پروژه با رسالت هدایت بهینه پروژه‌های خرد و کلان توسعه‌ای- عمرانی در سطوح بین‌المللی، ملی و منطقه‌ای و به منظور تحقق اهداف سه‌گانه آن‌ها یعنی زمان، بودجه و کیفیت، جایگاه ویژه‌ای در مهندسی، مدیریت و اقتصاد یافته است.

تاریخ وصول: ۹۴/۰۳/۲۰

تاریخ تصویب: ۹۵/۰۱/۲۲

راضیه چشمی ، کارشناس ارشد ریاضی کاربردی، دانشکده علوم ریاضی، دانشگاه فردوسی، آدرس ایمیل **ra.cheshomi@yahoo.com**

حمیدرضا یوسف زاده ، استادیار، دانشکده علوم ریاضی، گروه ریاضی کاربردی، دانشگاه پیام نور ایران، آدرس ایمیل **usefzadeh.math@pnu.ac.ir**

*__نویسنده مسئول مقاله:__ **حامدرضا طارقیان** ، استاد، دانشکده علوم ریاضی، دانشگاه فردوسی، آدرس ایمیل **taregian@um.ac.ir**

طراحی و پیاده‌سازی برنامه‌های عملکردی دانش محور که بتوانند در پرتو محدودیت‌های متعدد حاکم بر محیط اجرایی پروژه، مسایل گوناگونی را که در چرخه عمر پروژه روی می‌دهند به صورت بهینه حل و فصل کنند، ضرورتی اجتناب ناپذیر است. از جمله مهمترین این مسایل، مساله زمان‌بندی پروژه با منابع محدود است (در ادامه به این مساله با واژه RCPSP ارجاع می‌دهیم). در مسایل زمان‌بندی پروژه توابع هدف مختلفی در نظر گرفته می‌شود که می‌تواند جنبه پولی (مانند[۱]) و یا غیرپولی (مانند مقاله حاضر) داشته باشد. RCPSP مورد نظر در این مقاله بر طبق نمادگذاری *هرولن* و همکاران در زمره مسایل $m,1|cpm|C_{\max}$ قرار می‌گیرد [2]. در این مساله فرض است که هر فعالیت زمان اجرای ثابت و بدون وقفه دارد و اجرای هر فعالیت در گرو مصرف میزان مشخصی از منابع تجدیدپذیر است. هدف RCPSP زمان‌بندی فعالیت‌های پروژه به گونه‌ای است که ضمن رعایت محدودیت‌های منبع و پیش‌نیازی، زمان اجرای پروژه را کمینه کند. نسخه‌های متفاوتی از این مساله در ادبیات موضوع وجود دارند که بر اساس ماهیت فعالیت‌ها

(وقفه‌پذیر یا بدون وقفه، یک حالت اجرا یا چند حالت اجرا)، منابع موردنیاز (تجدیدپذیر و یا تجدیدناپذیر)، روابط پیش‌نیازی (یک رابطه‌ای یا چند رابطه‌ای)، نوع تابع هدف (زمان و یا هزینه)، تعداد تابع هدف و تعداد پروژه‌ها (تک پروژه‌ای یا چند پروژه‌ای) دسته‌بندی می‌شوند. یکی از این نسخه‌ها مساله زمان‌بندی پروژه با منابع محدود چندحالته است که در زمره مسایل $m,T|cpm,mu|C_{max}$ قرار دارد [2].

در مساله زمان‌بندی پروژه با منابع محدود چندحالته (در ادامه به این مساله با واژه MRCPSP ارجاع می‌دهیم)، هر فعالیت می‌تواند در حالت‌های مختلفی اجرا شود که برای هرکدام زمان و نیاز متفاوتی به منابع تعریف می‌شود. منابع مورد نیاز تنها به منابع تجدیدپذیر محدود نشده بلکه منابع تجدیدناپذیر و دوگانه نیز در نظر گرفته می‌شوند [3]. چون منابع دوگانه را می‌توان به‌عنوان ترکیبی از منابع تجدیدپذیر و تجدیدناپذیر در نظر گرفت، معمولا این دسته از منابع به‌طور صریح در نظر گرفته نمی‌شوند. هدف MRCPSPانتخاب یک حالت اجرایی برای هر فعالیت به‌منظور زمان‌بندی پروژه در زمان کمینه با رعایت تمام محدودیت‌هاست. چون این مساله تعمیم یافته RCPSP است لذا در دسته مسایل NP- سخت جای می‌گیرد. علاوه بر این، پیدا کردن جواب شدنی برای این مساله در شرایطی که فعالیت‌های پروژه بیش از یک منبع مصرف کنند، خود یک مساله NP- کامل است [4]. رویکردهای حل این مساله به دو دسته دقیق و ابتکاری تقسیم می‌شوند. به علت ماهیت پیچیده این مساله، روش‌های مبتنی بر رویکرد دقیق که یافتن جواب بهینه را تضمین می‌کنند به دلیل صرف زمان فراوان، کارآمد نیستند. به همین دلیل در ادبیات موضوع تنها تعداد اندکی روش‌های مبتنی بر این رویکرد ارایه شده است. به‌عنوان مثال *اسپرچر* [5] ، *هارتمن* و *درکسل*[6] ، *ژوو* همکارانش[7] از جمله محققانی هستند که روش‌هایی از این دست ارایه کرده‌اند.

روش‌های مبتنی بر رویکرد ابتکاری علی‌رغم این‌که حصول جواب بهینه مساله را تضمین نمی‌کنند اما برای حل مساله‌های پیچیده با ابعاد بزرگ به‌منظور تولید زمان‌بندی‌های نزدیک به بهینه در زمان کوتاه‌تر، از روش‌های مبتنی بر رویکرد دقیق کارآمدتر هستند. در اینجا به چند نمونه از روش‌های مبتنی بر این رویکرد در دو دسته ابتکاری و فراابتکاری اشاره شده است. در دسته روش‌های ابتکاری *اوزدامار* [8]، *لو* و همکاران [9] و در دسته روش‌های فراابتکاری *ون‌پتیقم* و *ونهوک* [10]، *یوسف‌زاده* و *طارقیان* [11] و *بشیکچی* و همکاران [12] از جمله محققانی هستند که روش‌هایی از این نوع ارایه کرده‌اند.

در سال ۲۰۱۱ *خولیو* و *ونهوک* روشی برای حل MRCPSP ارایه کردند. در این روش، آن‌ها به کمک حل‌کننده مساله صدق‌پذیری ابتدا MRCPSP را به یک RCPSP تبدیل کرده و سپس با استفاده از یکی از روش‌های ابتکاری به حل مساله تبدیل شده پرداختند. این روش در مقایسه با سایر روش‌های موجود برای حل MRCPSPاز قابلیت و کارایی بیشتری برخوردار است. با این وجود در حل بسیاری از مسایل دچار کمبود حافظه شده و زمان اجرای آن نیز غیرقابل قبول است [13].

در مقاله حاضر راهکارهایی برای بهبود *روش خولیو و ونهوک*از دو بعد زمان و حافظه پیشنهاد شده است. اعمال پیشنهادات ارایه شده سبب گردیده تا مشکل حافظه به لحاظ معیار تعداد عبارت‌های تضاد بالغ بر ۸۵ درصد و معیار طول متوسط عبارت تضاد بالغ بر ۱۵درصد و مشکل زمان به لحاظ معیارهای زمان اجرا و تعداد گره‌های ملاقات شده بالغ بر ۹۰ درصد بهبود یابد.

ادامه مقاله حاضر به صورت زیر ساماندهی شده است. در بخش ۲ روش *خولیو* و *ونهوک* تشریح و نقد می‌شود. در بخش ۳ راهکارهایی برای بهبود روش *خولیو* و *ونهوک* پیشنهاد می‌شود. در بخش ۴ با تحلیل پیشنهادات ارایه شده عملکرد آن‌ها ارزیابی می‌گردد. بخش پایانی مقاله به نتیجه‌گیری و ارایه پیشنهاد برای تحقیقات بیشتر اختصاص یافته است.

۲. ئروش خولیو و ونهوک

از ویژگی‌های روش حل *خولیوو ونهوک* برای MRCPSP آن است که تنها از یک لیست اولویت در هر نوبت اجرا استفاده می‌کند، گام‌ها در یک اجرای واحد انجام می‌شوند و پروژه‌ها با هر دو نوع منبع تجدیدپذیر و منبع تجدیدناپذیر قابل بررسی هستند. همین ویژگی‌ها موجب گردیده تا روش *خولیو* و *ونهوک* برای حل سایر نسخه‌های مساله زمان‌بندی پروژه با منابع محدود چندحالته نظیر $m,1T\,|cpm*,disc,mu|C_{\max}$ ، $m,1T\,|cpm,disc,id\,|C_{\max}$ و $m,1T\,|cpm,disc,id\,|other$ قابل تعمیم باشد [13]. مدل ریاضی MRCPSP در (۱) تا (۶) آمده است. در این مدل N مجموعه‌ای از فعالیت‌ها است. شماره‌گذاری فعالیت‌ها از یک فعالیت مجازی با شماره ۰ شروع و به یک فعالیت مجازی با شماره $n+1$ ختم می‌شود. در این مجموعه هر فعالیت غیر مجازی $i \in N$ در یکی از حالت‌های M_i با اطلاعات $(d_{i,m}, r^r_{i,m,k}, r^n_{i,m,l})$ که $m \in \{1,2,...,M_i\}$ اجرا می‌شود و شامل مدت زمان قطعی $d_{i,m}$ و متناظر آن $r^r_{i,m,k}$ واحد از منبع k و $r^n_{i,m,l}$ واحد از منبع l است. به منظور زمان‌بندی فعالیت‌ها یک مجموعه R^r از منابع تجدیدپذیر و یک مجموعه R^n از منابع تجدیدناپذیر در اختیار است. برای هر منبع $k \in R^r$ محدودیت دسترسی به میزان a^k_t در هردوره و برای هر منبع $l \in R^n$ محدودیت دسترسی به میزان a^n_l واحد در طی مدت زمان اجرای پروژه در نظر گرفته شده است. مدل ریاضیMRCPSP به شرح زیر است:

$$\text{Minimize} \sum_{t=es_{n+1}}^{ls_{n+1}} t x_{n+1,1,t} \qquad (۱)$$

subject to

$$\sum_{m=1}^{M_i}\sum_{t=es_i}^{ls_i}(t+d_{i,m})x_{i,m,t} \le \sum_{m=1}^{M_j}\sum_{t=es_j}^{ls_j} t x_{j,m,t} \quad \forall (i,j)\in A, \qquad (۲)$$

$$\sum_{m=1}^{M_i}\sum_{t=es_i}^{ls_i} x_{i,m,t} = 1 \qquad \forall i \in N \qquad (۳)$$

$$\sum_{i=1}^{n}\sum_{m=1}^{m_i} r_{i,m,k}^{r} \sum_{s=\max(t-d_{i,m,es_i})}^{\min(t-1,ls_i)} x_{i,m,s} \le a_k^r \quad \forall k\in R^r \ \text{and} \ t=1,...,T, \qquad (۴)$$

$$\sum_{i=1}^{n}\sum_{m=1}^{M_i} r_{i,m,l}^{n} \sum_{s=\max(t-d_{i,m,es_i})}^{ls_i} x_{i,m,t} \le a_k^n \quad \forall l\in R^n, \qquad (۵)$$

$$x_{i,m,t}\in\{0,1\} \ \forall i\in N;\ m=1,...,M_i;\ t=1,...,T \qquad (۶)$$

مدل (۱) تا (۶)، MRCPSP را به صورت یک مساله برنامه‌ریزی عدد صحیح با متغیر تصمیم $x_{i,m,t}$ (متغیر صفر و یک است)، فرمول‌بندی می‌کند، به‌طوری‌که اگر فعالیت i در زمان t با حالت اجرایی m $(1\le m\le M_i)$ انجام شود، $x_{i,m,t}=1$ و در غیر این صورت $x_{i,m,t}=0$ در نظر گرفته می‌شود. تابع هدف (۱) زمان اجرای پروژه را کمینه می‌کند. محدودیت (۲) یک رابطه پیش‌نیازی پایان به شروع با زمان صفر را در نظر می‌گیرد. محدودیت (۳) تضمین می‌کند تا هر فعالیت یک‌بار و در یک حالت و بدون وقفه اجرا شود. محدودیت (۴) تضمین می‌کند که مصرف منابع تجدیدپذیر در هر واحد زمان از موجودی این نوع منابع در هر واحد زمان فراتر نرود. محدودیت (۵) میزان نیاز به منابع تجدیدناپذیر را در ازای میزان موجودی این نوع منابع بررسی می‌کند. محدودیت (۶) متغیرهای تصمیم را وا می‌دارد تا ارزش صفر- یک اختیار کنند. متغیر T کران بالا برای اجرای پروژه است که مثلا می‌تواند برابر با مجموع بزرگترین زمان‌های اجرای فعالیت‌ها قرار بگیرد. با توجه به T می توان پنجره زمانی هر فعالیت را محاسبه کرد. برای هر فعالیت i زودترین زمان شروع es_i و دیرترین زمان شروع ls_i با استفاده از مسیر بحرانی و با در نظر گرفتن T و نیز کوچک‌ترین مدت‌زمان اجرای هر فعالیت تعیین می‌شود.

روش حل *خولیو* و *ونهوک* طی دو گام انجام می‌شود. در گام اول برای تامین محدودیت‌های منبع تجدیدناپذیر و انتخاب یک حالت اجرا برای هر فعالیت (یعنی صدق‌پذیر کردن محدودیت‌های (۳) و (۵)) محدودیت‌ها به شکل نرمال عطفی (CNF) درمی‌آیند. برای این منظور هر فعالیت چندحالته i (محدودیت (۳) را ببینید) به M_i زیرفعالیت تک‌حالته شکسته شده و سپس قید متناظر به صورت CNF در می‌آید. از طرفی هر محدودیت منبع تجدیدناپذیر (۵) که خود یک عبارت شبه *بولی* است با استفاده از یک درخت شمارشی به مجموعه‌ای از عبارت‌های CNF تبدیل می‌شود. عبارت‌های CNF تولید شده به انضمام لیستی از اولویت‌ها به حل‌کننده SAT وارد می‌شوند تا در نهایت مساله به یک RCPSP تبدیل شود. در گام دوم که گام زمان‌بندی است، RCPSP حاصل به کمک یکی از روش‌های حل ارایه شده در ادبیات موضوع حل می‌شود (به عبارت دیگر محدودیت‌های (۲) و (۴) صدق‌پذیر می‌شوند). لازم به ذکر است که از جمله روش‌های حل RCPSP روش ارایه شده در [14] است که سازوکار آن نیز مانند روش ارایه شده در این مقاله مبتنی بر حل کننده SAT است.

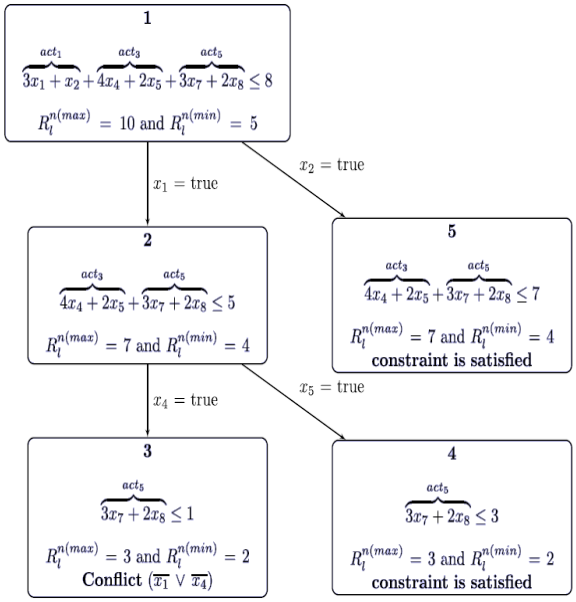
ورودی‌های حل‌کننده SAT که همان عبارت‌های CNF تولید شده در گام نخست هستند، ذخیره می‌شوند تا برای هر بار جستجو و به ازای هر لیست فعالیت، فراخوانی شوند. زمانی‌که اندازه پروژه براثر افزایش تعداد فعالیت‌ها و یا افزایش تعداد منابع تجدیدناپذیر بزرگ می‌شود، فضای جستجوی درخت شمارشی نیز وسیع‌تر گردیده و زمان بیشتری برای پیمایش آن مورد نیاز است. علاوه بر آن، تعداد عبارت‌های CNF حاصل از تبدیل عبارت شبه *بولی* محدودیت منبع تجدیدناپذیر نیز به صورت نمایی افزایش می‌یابد. تاثیر این افزایش‌ها ایجاد یک فایل ورودی حجیم برای حل‌کننده SAT است که نیاز بیش‌از اندازه آن را به حافظه درپی دارد. بزرگی فایل ورودی برای حل‌کننده SAT علاوه بر اشغال حافظه، موجب انجام محاسبات سنگین و درنتیجه صرف هزینه و زمان بسیاری می‌شود. در این مقاله برای جلوگیری از بروز این مشکلات، راهکارهایی پیشنهاد می‌شود. اما پیش از تبیین آن‌ها، ابتدا سازکار درخت شمارشی به طور مختصر تشریح می‌شود.

۱-۲. درخت شمارشی

مساله صدق‌پذیری و همچنین پیدا کردن جواب شدنی برای RCPSP وقتی فعالیت‌ها به دو یا بیشتر از دو منبع نیاز دارند جزو مسایل NP-کامل هستند [15]. از این رو، تعداد حالت‌های شدنی برای محدودیت منبع تجدیدناپذیر بسیار زیاد و شمارش همه این ترکیبات در عمل غیرممکن است. برای رفع این مشکل *خولیو* و *ونهوک* محدودیت منبع تجدیدناپذیر را با خارج کردن حالت‌های نشدنی، به یک نمونه مساله صدق‌پذیری تبدیل می‌کنند. چون هر فعالیت پروژه عملاً در یک حالت به اجرا در می‌آید، لازم است تا از میان حالت‌های مختلف اجرا برای هر فعالیت تنها یک حالت اجرا و به تبع آن یک میزان نیاز به منابع انتخاب شود. با استفاده از این اطلاعات و به کمک یک درخت شمارشی، محدودیت شبه *بولی* منبع تجدیدناپذیر به CNF تبدیل می‌شود.

در آغاز، درخت شمارشی، محدودیت منبع تجدیدناپذیر (مثلا محدودیت $3x_1+x_2+4x_4+2x_5+3x_7+2x_8\le 8$ در [13]) را به‌عنوان ورودی دریافت می‌کند. سپس در یک روند تکراری با تخصیص 1 به متغیرهای *بولی* (صفر و یک) x_i به‌تدریج اندازه محدودیت را کاهش می‌دهد. در هر سطح درخت شمارشی همه متغیرهای حالت مربوط به هر فعالیت بررسی می‌شوند و در آن سطح به ازای هر متغیری که 1 به آن اختصاص داده شده، یک گره ایجاد می‌گردد. بنابراین طی انجام این کار، اندازه محدودیت منبع تجدیدناپذیر به تدریج کاهش می‌یابد. ممکن است در این فرایند یک عبارت تضاد تولید شودکه برای ورود به حل‌کننده SAT ذخیره

مـی‌شـود. درخـت شمارشـی مربـوط بـه ایـن محـدودیت را در شکل۱ببینید.



شکل. ۱. درخت شمارشی [13]

حداقل و حداکثر نیاز فعالیت i به منبع تجدیدناپذیر l یعنی $r_{i,l}^{n(\min)}$ و $r_{i,l}^{n(\max)}$ به صورت زیر محاسبه می‌شوند:

$$r_{i,l}^{n(\min)}=\min_{m=i,\dots,M_i} r_{i,m,l}^{n} \quad , \quad r_{i,l}^{n(\max)}=\max_{m=i,\dots,M_i} r_{i,m,l}^{n} \qquad (۷)$$

ضمنا مجموع حداقل و حداکثر نیاز باقی‌مانده فعالیت i به منبع تجدیدناپذیر l یعنی $R_l^{n(\min)}$ و $R_l^{n(\max)}$ به صورت زیر محاسبه می‌شوند:

$$R_l^{n(\min)}=\sum_{i\in C} r_{i,l}^{n(\min)} \quad , \quad R_l^{n(\max)}=\sum_{i\in C} r_{i,l}^{n(\max)} \qquad (۸)$$

که در آن‌ها $C\subset N$ مجموعه‌ای از فعالیت‌هاست که در سطوح قبلی درخت انتخاب نشده‌اند (یعنی 1 به هیچ یک از حالت‌های اجرایی آن اختصاص نیافته است). با تعیین حالت هر فعالیت در هر سطح علاوه بر کوچک‌تر شدن عبارت، از میزان منبع موجود به اندازه نیاز فعالیت آن سطح کاسته شده و میزان منبع موجود به‌روز می‌شود (شکل ۱ را ببینید).

لازم است تا در هر گره محدودیت منبع تجدیدناپذیر باقی‌مانده ارزیابی شود تا مشخص گردد که آیا شاخه هرس می‌شود یا خیر؟ و در صورت هرس شدن، پسگرد با اضافه کردن عبارت تضاد همراه است یا خیر؟ برای بررسی این حالت در هر گره، دو قانون ارزیابی در نظر گرفته شده است که به‌صورت زیر تعریف می‌شوند [13]:

۱. اگر رابطه

$$R_l^{n(\max)}\le a_l^n(rem) \qquad (۹)$$

برقرار باشد، محدودیت باقی‌مانده منبع تجدیدناپذیر صدق‌پذیر می‌شود، زیرا اگر همه فعالیت‌ها در حالتی اجرا شوند که بیشترین نیاز به منبع را داشته باشند، مقدار منبع باقی‌مانده برای آن‌ها کفایت می‌کند. در این حالت با پسگرد، کار با گره(های) دیگر در همان سطح از درخت و بدون اضافه کردن عبارت تضاد ادامه می‌یابد. مثلاًدر گره ۵ درخت شمارشی شکل ۱، رابطه‌ی($R_l^{n(\max)}=7\le 7=a_l^n(rem)$) حاکی از آن است که محدودیت باقی‌مانده صدق‌پذیر است و الگوریتم پسگرد خواهد داشت.

۲. اگر رابطه

$$R_l^{n(\min)}>a_l^n(rem) \qquad (۱۰)$$

برقرار باشد، صدق‌پذیر شدن محدودیت منبع تجدیدناپذیر باقی‌مانده غیر ممکن است، زیرا حتی اگر همه فعالیت‌ها در حالتی اجرا شوند که کمترین نیاز به منبع را داشته باشند، مقدار منبع باقی‌مانده برای آن‌ها کفایت نمی‌کند. در این حالت روند حل به تناقض رسیده است. یک عبارت تضاد که حاوی نقیض همه حالت‌هایی است که در سطوح قبلی سطح جاری درخت انتخاب شده‌اند، تولید شده و سپس پسگرد انجام می‌شود و کار با گره(های) دیگر در همان سطح از درخت ادامه می‌یابد. همان‌طور که در شکل۱مشاهده می‌شود، نمودار شمارشی در گره ۳ به تناقض رسیده است. بنابراین عبارت تضاد آن مسیر یعنی $(\bar{x}_1\vee\bar{x}_4)$ تولید می‌شود و روند الگوریتم با بررسی گره ۴ ادامه می‌یابد.

به‌منظور بهبود کارایی پیمایش درخت شمارشی، در روش *خولیو* و *ونهوک* دو قانون بهبوددهنده ساده برای کاهش تعداد گره‌ها در نظر گرفته شده که به شرح زیرند:

۱. اگر همه حالت‌های یک فعالیت، نیاز یکسانی برای منبع تجدیدناپذیر داشته باشند، می‌توانند از جستجو حذف شوند و نیاز آن فعالیت از کل موجودی منبع کاسته شود.

۲. اگر عبارت

$$r_{i,m,l}^{n}-r_{i,l}^{n(\min)}>a_l^n(rem)-R_l^{n(\min)} \qquad (۱۱)$$

برقرار باشد، آنگاه حالت m ام اجرای فعالیت i می‌تواند معادل 0 شده و از جستجو حذف شود. برای مثال اگر این بررسی را در گره ۱ انجام دهیم، حاصل عبارت $r_{i,m,l}^{n}-r_{i,l}^{n(\min)}$ برای همه حالت‌های اجرای فعالیت‌های۱ تا ۵ شکل ۱ به ترتیب برابر ۲، ۰، ۰، ۲، ۰، ۰، ۱ و ۰ است که مقایسه آن اعداد با $a_l^n(rem)-R_l^{n(\min)}=8-5=3$ بیانگر این واقعیت است که هیچ یک از فعالیت‌ها از جستجو حذف نخواهند شد. اکنون محدودیت منبع تجدیدناپذیر با عبارت $(\bar{x}_1\vee\bar{x}_4)$ به‌صورت CNF تبدیل‌شده و آماده‌ی ورود به حل‌کننده SATاست.

۳. بهبود الگوریتم پیمایش درخت شمارشی

به منظور کاهش میزان نیاز بـه حافظـه در ذخیـره‌سـازی خروجـی درخت شمارشی، پیشنهاد شده تا به جای تبدیل محـدودیت شـبه *بولی* منبع تجدیدناپذیر به‌صورت CNF، این محدودیت به کمک دو قانون ارزیابی (رابطه‌های (۹) و (۱۰)) و دو قـانون بهبوددهنـده بـه صورت پویا و مستقیم به حل‌کننده SAT وارد شود تـا طـی اجـرای حل‌کننده، علاوه بر انتخـاب یـک حالـت اجـرا بـرای هـر فعالیـت، محدودیت منبع تجدیدناپذیر نیز مورد بررسی قرار گیرد[13]. نتایج گزارش شده حاکی از آن است که ایـن روش میـزان حافظـه مـورد نیاز و تعداد عبارت‌های محدودیت را کاهش می‌دهد. با وجود آنکـه کاهش حافظه تاثیر مثبتی بر اختصاص حافظه اولیه در زمـان اجـرا دارد، اما موجب افزایش سرعت ادامه روند در جستجوی حل کننده نمی‌شود زیرا تعداد گام‌های الگـوریتم را تغییـر نمـی‌دهـد و نتـایج یکسان است.

در این مقاله، با اضافه کردن گام‌های جدید بـه الگـوریتم پیمـایش درخت شمارشی (مرتب‌سازی و دو قاعده هرس)، مشـکل حافظـه و کمبود زمان بهبود داده می‌شود. به دو دلیل توجه اصلی ایـن مقالـه بر بهسازی الگوریتم پیمایش درخت شمارشی معطوف شـده اسـت. اول آن که، مساله صدق‌پذیری خود یک مساله NP-کامل است کـه سالانه طی مسابقات بین‌المللی روش‌های کارآمدتری برای حـل آن ارایه می‌شود لذا فرض بر این است که در روند حل، یک حل‌کننـده کارآمد در اختیار است. دوم آن که، RCPSP نیز دارای ادبیات غنـی و مشحون از روش‌های حل دقیق و ابتکاری اسـت. از ایـن رو، ایـن مقاله بر بهبود الگوریتم پیمـایش درخـت شمارشـی متمرکـز شـده است.

درادامه هر یک از گام‌هـای اضـافه شـده بـه روش، تشـریح شـده و سپس مزایای هر کدام تبیین می‌شوند. چـون در ایـن بخـش تنهـا درباره بهبود گـام اول الگـوریتم کـه مربـوط بـه محـدودیت منبـع تجدیدناپذیر می‌باشد، بحث مـی‌شـود لـذا در ادامـه منظـور از واژه منبع، منبع تجدیدناپذیر است.

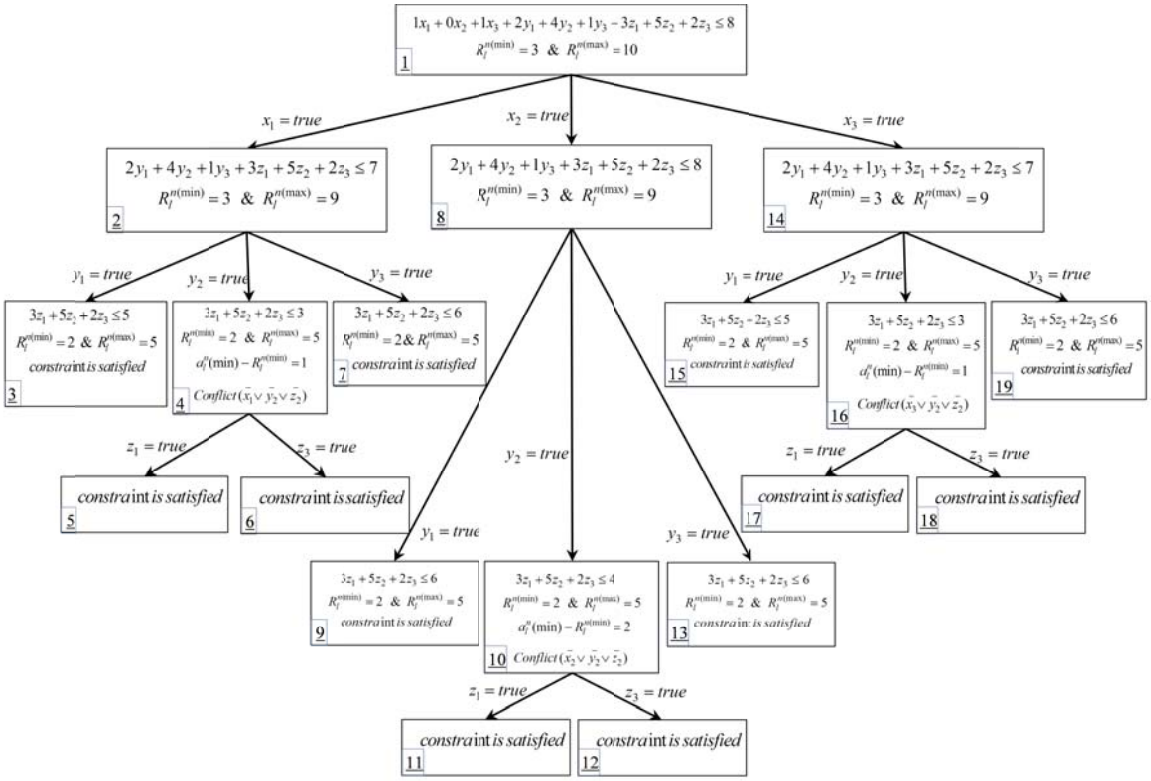
۱-۳. مرتب‌سازی اولیه

درMRCPSP هر فعالیت دارای چند حالت اجراست که هر حالت به میزان منبع مشخصی نیاز دارد. برای اینکه یک فعالیـت اجـرا شـود باید حداقل نیازش به منبع برآورده شود. لذا می‌توان در همان ابتدا حداقل نیاز هر فعالیت را از کل موجودی منبع کسـر کـرد و بـه آن فعالیـت اختصـاص داد تـا از انجـام محاسـبات اضـافی در ادامـه جلوگیری شود.

هـدف اصـلی از تبـدیل عبـارت شـبه *بـولی* بـه CNF در درخـت شمارشی، خارج کردن عبارت‌های تضاد اسـت. بنـابراین تسـریع در خارج کردن عبارت‌های تضاد، موجب صرفه‌جویی در زمان پـردازش می‌شود. درپیمایش درخت شمارشی، هنگامی تضاد رخ می‌دهد که مجموع حداقل نیاز هر فعالیت به منبع از میزان موجودی آن منبـع تجاوز کند (یا $R_l^{n(\min)} > a_l^n(rem)$). بنابراین، برقراری این رابطـه در سطوح بالاتر درخت به کارایی بیشتر منجر خواهـد شـد. پـس لازم می‌شود تا فعالیت‌هایی در سـطوح بـالاتر درخـت قـرار بگیرنـد کـه حـداکثر میـزان نیـاز بـه منبـع را دارنـد تـا بدینوسـیله موجـودی باقی‌مانده منبع هر چه سریع‌تر از حداقل نیاز سـایر فعالیـت‌هـا بـه منبع کمتر شود و تضاد رخ دهد.

از این رو، عبارت ورودی باید طوری مرتب شود که فعالیت‌هایی که نیاز بیشتری به منبع دارند زودتر مورد بررسی قرارگیرند. بـا توجـه به آنچه ذکر شد گام‌های مرتب‌سازی اولیه برای عبارت شـبه *بـولی* $1x_1 + 0x_2 + 1x_3 + 2y_1 + 4y_2 + 1y_3 + 3z_1 + 5z_2 + 2z_3 \le 8$ کـــــه مبین محدودیت منبع در پروژه‌ای با سه فعالیت x، y و z است کـه هر یک از این فعالیت‌ها می‌توانند دریکی از سه حالت اجرا شوند، به شرح زیر می‌باشد.

در شکل ۲ درخت شمارشی عبارت شـبه *بـولی* مـورد نظـر قبـل از مرتب‌سازی نمایش داده شده است.



شکل. ۲. درخت شمارشی پیش از عملیات مرتب‌سازی

مرتب‌سازی به روال زیر انجام می‌شود:

گام۱. (کاهش حداقل نیاز): حداقل نیاز به منبع هر فعالیت از نیاز به منبع هر حالت همان فعالیت و مجموع حداقل نیاز به منبع هر فعالیت از موجودی منبع که در این مثال به ترتیب $R_l^{n(min)} = 3$ و $a_l^n = 8$ است، کسر می‌شود. به این ترتیب حداقل نیاز منبع صفر خواهد شد.

$$1x_1 + 0x_2 + 1x_3 + 2y_1 + 4y_2 + 1y_3 + 3z_1 + 5z_2 + 2z_3 \le 8 \quad \Rightarrow \quad 1x_1 + 0x_2 + 1x_3 + 1y_1 + 3y_2 + 0y_3 + 1z_1 + 3z_2 + 0z_3 \le 5$$

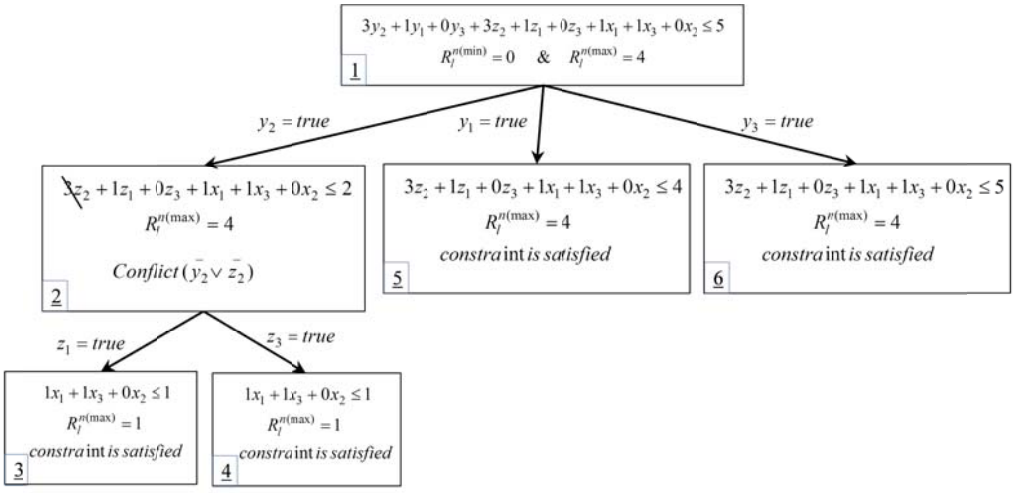
گام۲. (مرتب سازی حالت‌ها): حالت‌های مختلف به ترتیب نزولی میزان نیاز به منبع مورد نظر برای هر فعالیت مرتب می‌شوند.

$$1x_1 + 0x_2 + 1x_3 + 1y_1 + 3y_2 + 0y_3 + 1z_1 + 3z_2 + 0z_3 \le 5 \quad \Rightarrow \quad 1x_1 + 1x_3 + 0x_2 + 3y_2 + 1y_1 + 0y_3 + 3z_2 + 1z_1 + 0z_3 \le 5$$

گام۳. (مرتب‌سازی فعالیت‌ها): براساس اولین حالت هر فعالیت و یا به عبارتی دیگر با توجه به گام قبل براساس حالتی که بیشترین نیاز به منبع را دارد، فعالیت‌ها به ترتیب نزولی مرتب می‌شوند.

$$1x_1 + 1x_3 + 0x_2 + 3y_2 + 1y_1 + 0y_3 + 3z_2 + 1z_1 + 0z_3 \le 5 \quad \Rightarrow \quad 3y_2 + 1y_1 + 0y_3 + 3z_2 + 1z_1 + 0z_3 + 1x_1 + 1x_3 + 0x_2 \le 5$$

در شکل ۳ درخت شمارشی عبارت مورد نظر پس از مرتب‌سازی نمایش داده شده است.



شکل. ۳. درخت شمارشی پس از عملیات مرتب‌سازی

با مقایسه‌ی دو درخت شمارشی در شکل‌های ۲ و ۳، مشاهده می‌شود که تعداد شاخه‌ها، گره‌ها و عبارت‌ها و نیز اندازه عبارت‌ها با انجام عملیات مرتب‌سازی اولیه کاهش چشم‌گیری داشته است. در بخش ۴، چگونگی و میزان این کاهش بر روی مسایل استاندارد مورد تجزیه و تحلیل قرار می‌گیرد.

۱-۱-۳. بررسی عملکرد مرتب‌سازی

تاثیر مثبت مرتب‌سازی بر الگوریتم پیمایش درخت شمارشی از دو جنبه‌ی زمان و حافظه مورد توجه است:

الف. کاهش زمان

۱. با انجام مرتب‌سازی و چینش فعالیت‌ها به ترتیب نزولی نسبت به میزان نیاز به منبع، موجودی منبع در سطوح بالاتر درخت مصرف می‌شود و جستجو سریع‌تر به تضاد می‌رسد. این وضعیت در مثال ذکر شده نیز مشهود است زیرا پیش از مرتب سازی، اولین عبارت تضاد در گره شماره ۴ و پس از مرتب‌سازی در گره شماره ۲ یافت شده است.

۲. به علت کاهشی که در گام ۱ مرحله مرتب‌سازی اتفاق می‌افتد، تعداد محاسبات در گره‌ها کاهش می‌یابد. در مرحله‌ی مرتب‌سازی حداقل نیاز هر فعالیت به منبع به آن فعالیت اختصاص می‌یابد پس $R_l^{n(min)}$ در تمامی گره‌ها برابر صفر می‌شود. هم‌چنین آنچه به عنوان نیاز به منبع برای هر حالت اجرا باقی می‌ماند، معادل عبارت $r_{i,m,l}^n - r_{i,l}^{n(min)}$ و میزان دسترسی به منبع معادل عبارت $a_l^n(rem) - R_l^{n(min)}$ در قاعده بهبود دهنده۲ (یعنی رابطه ۱۱) است. پس برای بررسی نامساوی $r_{i,m,l}^n - r_{i,l}^{n(min)} > a_l^n(rem) - R_l^{n(min)}$ به منظور اجرای گام بهبود دهنده۲ در هر گره نیاز به محاسبات اضافی نیست و تنها کافی است هر یک از ضریب متغیرها با عدد طرف دیگر عبارت شبه *بولی* مقایسه شود.

۳. پس از مرتب‌سازی، تعداد گره کمتری برای بررسی فضای جستجو پیمایش می‌شود و این اتفاق به معنای شاخه‌سازی کمتر و در نتیجه صرفه‌جویی بیشتر زمان می‌باشد. به‌طوری که ملاحظه می‌شود درخت شمارشی مثال مورد نظر پیش از عملیات مرتب‌سازی ۱۹ گره دارد که این تعداد پس از مرتب‌سازی به ۶ گره رسیده است.

ب. حافظه

۱. چون مرتب‌سازی زمینه را برای یافتن تضاد در سطوح بالاتر درخت شمارشی فراهم می‌سازد، باعث می‌شود تا از میزان ورود متغیرهایی که در محدودیت منبع تضاد ایجاد نمی‌کنند، کاسته شود. بنابراین پس از مرتب‌سازی، عبارت تضاد غالباً عوامل اصلی ایجاد تضاد را در بر می‌گیرد و عبارت کوچک‌تری را به نسبت پیش از مرتب‌سازی تولید می‌کند.

۲. به علت وارد نشدن متغیرهای اضافی به عبارت تضاد از تکرار متغیرهای اصلی تضاد با متغیرهای اضافی جلوگیری می‌شود و تعداد عبارت‌ها کاهش می‌یابد. در مثالی که ذکر شد عامل اصلی ایجاد تضاد دو متغیر y_2 و z_3 است که در درخت شمارشی پس از مرتب‌سازی، به طور دقیق در عبارت تضاد آمده‌اند. در حالی که همین دو متغیر پیش از مرتب‌سازی در سطوح پایین‌تر و با دخالت سه متغیر در سه عبارت تضاد مجزای $(\bar{x}_1 \vee \bar{y}_2 \vee \bar{z}_2)$، $(\bar{x}_2 \vee \bar{y}_2 \vee \bar{z}_2)$ و $(\bar{x}_3 \vee \bar{y}_2 \vee \bar{z}_2)$ آمده‌اند. کوچک شدن و کم شدن تعداد عبارت‌های تضاد موجب صرفه‌جویی در اشغال حافظه و کاهش حجم ورودی به حل‌کننده SAT می‌شود.

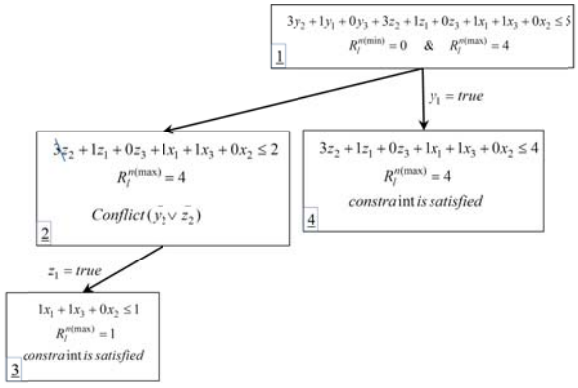
۲-۳. قاعده هرس۱ (پیش‌بینی صدق‌پذیری)

اگر در یک گره، مجموع حداکثر نیاز فعالیت‌ها به یک منبع از موجودی آن منبع نابیشتر باشد (به عبارت دیگر رابطه (۹) یعنی $R_l^{n(max)} \le a_l^n(rem)$ برقرار باشد)، آنگاه محدودیت منبع در آن

شاخه صدق‌پذیر و در نتیجه شاخه هرس می‌شود. پس از هرس، الگوریتم پسگرد انجام می‌دهد و در صورت تمام نشدن حالت‌های فعالیت برای شاخه‌سازی، یک حالت دیگر از همان فعالیت را انتخاب و در همان سطح یک گره دیگر ایجاد می‌کند.

با اضافه کردن مرتب‌سازی به الگوریتم، حالت‌های مختلف هر فعالیت برحسب نیاز آن به منابع به ترتیب نزولی مرتب شده و سپس مورد بررسی قرار می‌گیرند. با این توصیف اگر تاثیر این مرتب‌سازی بر باقی‌مانده موجودی منبع در هر گره از یک سطح مورد ارزیابی قرار گیرد، این نتیجه حاصل می‌شود که به علت ترتیب نزولی در بررسی حالت‌ها، با پیش رفتن در گره‌های یک سطح به تدریج مصرف منبع کاهش می‌یابد. به این ترتیب چون در هر سطح مجموع حداکثر نیاز به منبع فعالیت‌ها ($R_l^{n(max)}$) ثابت است بنابراین اگر در گرهی قانون ارزیابی $R_l^{n(max)} \le a_l^n(rem)$ برقرار شود، به علت ترتیب صعودی باقی‌مانده موجودی منبع، این رابطه قطعاً در گره‌های بعدی نیز برقرار خواهد بود. بنابراین با توجه به آنچه شرح داده شد صدق‌پذیری گره‌ها در یک سطح قابل پیش‌بینی است و با صدق‌پذیر شدن یک گره با قطعیت و بدون انجام هیچ‌گونه محاسبات اضافی می‌توان نتیجه گرفت که گره‌های بعدی آن سطح نیز صدق‌پذیر و لذا قابل هرس کردن هستند.

به‌طوری‌که ملاحظه شد درخت شمارشی شکل ۲ پس از مرتب‌سازی به درخت شمارشی شکل ۳ تبدیل شد. حال اگر قاعده هرس۱ روی آن پیاده شود، بدون هیچ محاسبه‌ای صدق‌پذیری دو گره پیش‌بینی می‌شوند و درخت به شکل ۴ تبدیل می‌شود.



شکل. ۴. درخت شمارشی پس از اجرای قاعده هرس۱

۳-۲-۱. بررسی عملکرد قاعده هرس۱

قاعده هرس۱ مانع ایجاد گره و انجام محاسبات اضافی در گره‌هایی که صدق‌پذیری آن‌ها قابل پیش‌بینی است، می‌شود. در شکل ۳ صدق‌پذیری دو گره شماره ۴ و گره شماره ۶ با توجه به قاعده مطرح شده قابل پیش‌بینی است لذا دیگر مورد بررسی قرار نمی‌گیرند و تعداد گره‌ها از ۶ گره به ۴ گره کاهش می‌یابد (شکل ۴ را ببینید).

۳-۳. قاعده هرس۲ (محاسبه بیشینه دوم)

در بخش ۱-۲ برای بررسی هر گره دو قانون ارزیابی معرفی شد. قانون اول مجموع حداکثر نیاز به منبع فعالیت‌ها را با باقی‌مانده موجودی منبع مقایسه می‌کند ($R_l^{n(\max)} \le a_l^n(rem)$) تا اگر حداکثر نیاز به منبع فعالیت از موجودی منبع کمتر باشد، نتیجه بگیرد که مشکلی در تامین منبع در ادامه بررسی وجود ندارد. لذا آن شاخه صدق‌پذیر است و نیازی به ادامه بررسی نیست. در قانون دوم حالت اجرایی m فعالیت i در صورت برقراری رابطه ($r_{i,m,l}^n - r_{i,l}^{n(min)} > a_l^n(rem) - R_l^{n(min)}$) از جستجو حذف می‌شود. در این بخش با استفاده از این دو قانون یک قاعده هرس جدید پیشنهاد می‌شودکه در اثر آن فضای جستجو کاهش می‌یابد.

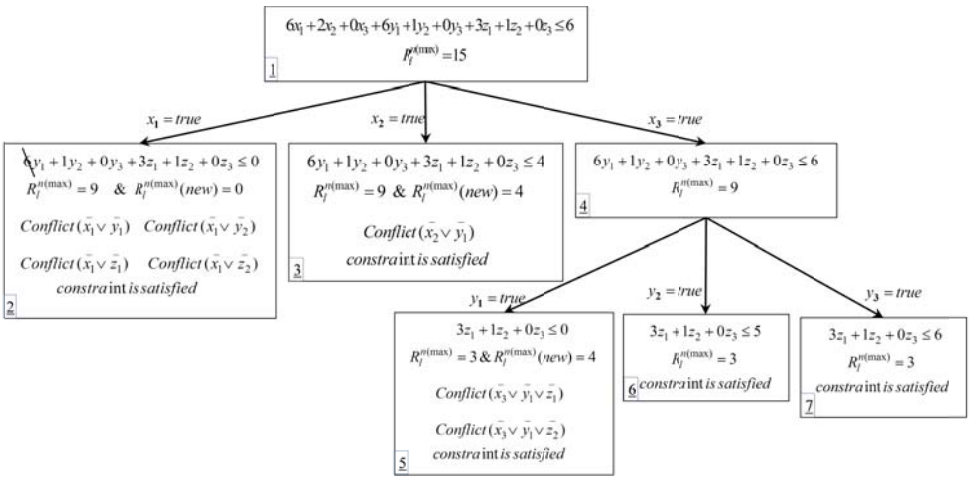
شکل ۵ را که در آن درخت شمارشی متناظر با عبارت مرتب شده $6x_1+2x_2+0x_3+6y_1+1y_2+0y_3+3z_1+1z_2+0z_3 \le 6$ (به محدودیت منبع پروژه‌ای با سه فعالیت که هر کدام دارای سه حالت اجرا هستند مربوط می شود)، نشان داده شده ببینید.

همانطور که در شکل ۵ مشاهده می‌شود در بعضی از گره‌ها پس از اعمال قانون بهبوددهنده۲ (رابطه (۱۱)) تعدادی از حالت‌ها حذف شده‌اند. طبق این رابطه اگر قرار باشد حالت‌هایی از فعالیت حذف شوند، انتظار می‌رود آن حالت‌هایی از فعالیت باشند که نیاز بیشتری به منبع دارند، لذا با حذف حتی یک حالت از فضای جستجو، حداکثر نیاز به منبع آن فعالیت تغییر می‌کند. این تغییر موجب می‌شود تا فرصتی برای بررسی مجدد رابطه (۹) (یعنی $R_l^{n(max)} \le a_l^n(rem)$) و هرس شاخه فراهم شود.

به این ترتیب در هر گره پس از حذف حالت‌هایی که در قانون بهبوددهنده۲ صادق هستند، مجدداً مجموع حداکثر نیاز به منبع فعالیت‌ها محاسبه می‌شود ($R_l^{n(max)}(new)$). در این مرحله برای بررسی صدق‌پذیری لازم است تا رابطه (۱۲) برقرار باشد.

$$R_l^{n(max)}(new) \le a_l^n(rem) \qquad (۱۲)$$

در صورت صادق بودن رابطه (۱۲) شاخه هرس می‌شود. شکل ۶ درخت شمارشی مربوط به عبارت مورد نظر را پس از اعمال قاعده هرس۱ نمایش می‌دهد. همانطور که ملاحظه می‌شود در گره شماره ۲ با محاسبه‌ی $R_l^{n(max)}(new)$ برای رسیدن به صدق‌پذیری و هرس گره، دیگر نیازی به ایجاد زیر سطح جدید نیست.



شکل. ۵. درخت شمارشی پس از اجرای قاعده هرس۲

۱-۳-۳. بررسی عملکرد قاعده هرس۲

در صورتی که رابطه (۱۲) برقرار باشد، از ادامه عملیات جستجو در عمق جلوگیری می‌شود و هرس در همان سطح‌های اولیه صورت می‌گیرد که این امر موجب می‌شود تعداد گره‌های ملاقات شده به طور چشمگیری کاهش یابد. در مثال مورد نظر تعداد گره‌های ملاقات شده از ۱۱ گره به ۷ گره کاهش یافته است.

۳-۴. تفاوت صدق‌پذیری‌ها

برای بررسی صدق‌پذیری یک شاخه علاوه بر بررسی رابطه (۱۰)، رابطه (۱۲) نیز مورد بررسی قرار می‌گیرد. این دو صدق‌پذیری از دیدگاه قاعده هرس۱ متفاوتند زیرا قاعده هرس۱ از روی گره‌هایی که حاوی عبارت تضاد نیستند و با قانون ارزیابی به صدق‌پذیری رسیده‌اند می‌تواند صدق‌پذیری گره‌های بعدی را پیش‌بینی کند، ولی اگر صدق‌پذیری گره، حاصل از قاعده هرس۲ باشد، به علت وجود تضاد در آن گره هیچ تضمینی برای صدق‌پذیری گره‌های بعدی وجود ندارد. برای درک بهتر این موضوع لازم است به گره شماره ۳ درخت شمارشی شکل ۶ توجه کنید. به‌طوری‌که ملاحظه می‌شود با توجه به صدق پذیری این گره با قاعده‌ی هرس۱ نمی‌توان صدق‌پذیری گره شماره ۴ را نتیجه گرفت زیرا به‌طوری که ملاحظه می‌شود گره بعدی در این سطح صدق‌پذیر نیست.

۴. آزمون‌ها وتجزیه و تحلیل نتایج

در بخش ۲ به مشکلات موجود در مرحله استفاده از درخت شمارشی در روش حل MRCPSP با استفاده از حل‌کننده SAT اشاره شد. در بخش پیشنهاد شد تا به منظور بهبود روش حل *خولیو* و *ونهوک*، مرتب‌سازی اولیه و دو قاعده هرس به الگوریتم پیمایش درخت شمارشی پایه (یعنی الگوریتم پیمایش درخت شمارشی مطرح شده در بخش ۱-۲) اضافه شود.

در این بخش نتایج حاصل از ارزیابی عملکرد الگوریتم پیمایش درخت شمارشی در حالت پایه و نیز در حالت‌هایی که مرتب‌سازی اولیه و قاعده‌های هرس به آن اضافه شده‌اند، گزارش داده می‌شود. به جهت تحلیل عملکرد حالت‌های مختلف، معیارهایی تعیین شده تا با مبنا قرار دادن آن‌ها، حالت‌های مختلف اجرا شده و سپس با هم مقایسه شوند تا مشخص شود که بخش‌های اضافه شده به الگوریتم پایه تا چه حد در بهبود روند الگوریتم موثر بوده‌اند. به‌منظور ارزیابی تاثیر راهکارهای پیشنهاد شده بر زمان اجرا و هم‌چنین میزان حافظه مورد نیاز پنج سناریوی مختلف بر روی مسایل کتابخانه‌ای استاندارد *کولیش* اجرا شده است.

۱-۴. سناریوهای مورد آزمون

پنج سناریوی مورد آزمون در ذیل شرح داده شده است:

۱. الگوریتم پایه: اجرای الگوریتم پیمایش درخت شمارشی که در بخش ۱-۲ مطرح شده است.
۲. مرتب‌سازی: اجرای الگوریتم پیمایش درخت شمارشی با اضافه کردن گام مرتب‌سازی اولیه مطابق آنچه در بخش ۱-۳ مطرح شده است، (الگوریتم پایه با مرتب‌سازی اولیه).
۳. هرس۱: اجرای الگوریتم پیمایش درخت شمارشی پس از مرتب‌سازی اولیه و اعمال قاعده هرس۱ مطابق آنچه در بخش مطرح شده است، (الگوریتم پایه با مرتب‌سازی اولیه و هرس۱).
۴. هرس۲: نتایج حاصل از اجرای الگوریتم پیمایش درخت شمارشی پس از مرتب‌سازی و اعمال قاعده هرس۲ مطابق آنچه در بخش۳-۳ مطرح شده است، (الگوریتم پایه با مرتب‌سازی اولیه و هرس۲).
۵. الگوریتم نهایی: اجرای الگوریتم پیمایش درخت شمارشی پس از مرتب‌سازی و اعمال هر دو قاعده هرس به طور هم‌زمان، به منظور اینکه تاثیر قواعد پیشنهاد شده به طور هم‌زمان مورد بررسی قرار گیرد، (الگوریتم پایه با مرتب‌سازی اولیه، هرس۱ و هرس۲).

۴-۲. محیط اجرایی آزمون‌ها

آزمون‌ها با نرم‌افزار *متلب* کدنویسی شده‌اند. چون نیازی به مقایسه عملکرد نمونه‌های مختلف با یکدیگر نیست و فقط مقایسه عملکرد شیوه‌های مختلف بر روی یک نمونه مد نظر است لذا برای تسریع در اجرا از چند کامپیوتر به طور همزمان استفاده شده است.

۴-۳. معیارها

زمان و حافظه دو مشکل اصلی روش حل ارایه شده برایMRCPSP هستند. از این‌رو با تعیین معیارهایی که روایی دارند، پیشنهادات مطرح شده برای بهبود روند پیمایش درخت شمارشی مورد ارزیابی قرار می‌گیرند.

در هر آزمایش دو معیار برای اندازه‌گیری حافظه و دو معیار برای اندازه‌گیری زمان به شرح زیر مد نظر بوده است:

الف. حافظه:

۱. تعداد عبارت‌های تضاد: عبارت‌های تضاد که نتیجه تبدیل عبارت شبه *بولی* مربوط به محدودیت منبع تجدیدناپذیر به CNF است، باید به‌عنوان ورودی حل‌کنندهSAT در حافظه ذخیره شوند. هر چه تعداد این عبارت‌ها افزایش یابد فضای بیشتری را در حافظه اشغال می‌کند. رشد نمایی تعداد عبارت‌ها به علت افزایش تعداد منابع و یا تعداد فعالیت‌ها، منجر به یک فایل ورودی حجیم برای حل‌کننده‌SAT و در نتیجه بروز مشکل حافظه می‌شود. به همین علت برای سنجش حافظه از معیار تعداد عبارت‌های تضاد استفاده شده است.

۲. طول متوسط هر عبارت تضاد: به تعداد متغیرهای هر عبارت تضاد، طول عبارت گویند. هر چه طول عبارت بیشتر باشد، عبارت حافظه بیشتری اشغال می‌کند و ممکن است موجب کمبود حافظه شود. بنابراین در هر آزمون، نسبت تعداد کل متغیرهای موجود در عبارت‌های تضاد به تعداد عبارت‌های تضاد که در واقع نشانگر طول متوسط هر عبارت تضاد است، می‌تواند به‌عنوان یک معیار برای سنجش میزان حافظه اشغال شده مورد استفاده قرار گیرد.

ب. زمان:

۱. زمان اجرا: اندازه‌گیری زمان اجرای هر آزمون بر حسب ثانیه معیار رایجی برای سنجش زمان است.

۲. گره‌های ملاقات شده: در یک درخت هر چه تعداد گره‌ها افزایش یابد زمان بیشتری برای بررسی فضای جستجو صرف می‌شود. بنابراین تعداد گره‌های ملاقات شده معیار دیگری برای سنجش زمان است.

۴-۴. اعتبارسنجی

در این مقاله با اضافه کردن راهکارهای مرتب‌سازی و دو قاعده‌ی هرس به الگوریتم پیمایش درخت شمارشی، بهبود عملکرد روش *خولیو* و *ونهوک* برای حلMRCPSP مورد مطالعه و بررسی قرار گرفته است. برای تعیین اعتبار الگوریتم بهبودیافته حاصل به همراه حل کننده‌SAT و تایید این که خروجی این دو با هم مسایل شدنی RCPSP است. تعدادی از مسایل نمونه‌ای *کولیش* به تصادف انتخاب و الگوریتم پیشنهادی بر روی آن‌ها اجرا شده است. برای اجتناب از طولانی شدن مقاله نتایج کامل اعتبارسنجی در آدرس اینترنتی قرار داده شده است. جدول۱ نمونه کوچکی از این اجراها را نشان می‌دهد. همان‌گونه که ملاحظه می‌شود، مثلاً در مورد مساله نمونه J106_3 (پروژه‌ای با ۱۰ فعالیت که هر فعالیت سه حالت اجرایی دارد) در ازای عبارت ورودی، الگوریتم یک RCPSP تولید می‌کند، (هر فعالیت مساله یک حالت اجرایی و در نظر گفتن منابع تجدیدناپذیر) به گونه‌ای که محدودیت منابع تجدیدناپذیر متناظر آن نیز شدنی است. همین روند برای سایر نمونه های مورد آزمون نیز برقرار است.

جدول. ۱. اعتبار سنجی الگوریتم پیشنهادی پیمایش درخت شمارشی

نام نمونه	حاصل اجرای الگوریتم پیشنهادی پیمایش درخت	عبارت ورودی ($x_{i,j}$ متغیر *بولی* مربوط به حالت j فعالیت i است)
	RCPSP تولید شده:	$7x_{1,1}+6x_{1,2}+5x_{1,3}+9x_{2,1}+6x_{2,2}+1x_{2,3}+5x_{3,1}+4x_{3,2}+2x_{3,3}+9x_{4,1}+9x_{4,2}$
	$\{x_{1,1},x_{2,1},x_{3,1},x_{4,1},x_{5,1},$	$+9x_{4,3}+8x_{5,1}+4x_{5,2}+4x_{5,3}+9x_{6,1}+7x_{6,2}+2x_{6,3}+9x_{7,1}+9x_{7,2}+7x_{7,3}$
	$x_{6,3},x_{7,3},x_{8,1},x_{9,3},x_{10,3}\}$	$+10x_{8,1}+10x_{8,2}+10x_{8,3}+7x_{9,1}+6x_{9,2}+3x_{9,3}+6x_{10,1}+9x_{10,2}+6x_{10,3}\le 66$
J106_3	بررسی شدنی بودن RCPSP تولید شده:	$10x_{1,1}+7x_{1,2}+7x_{1,3}+7x_{2,1}+6x_{2,2}+6x_{2,3}+7x_{3,1}+8x_{3,2}+6x_{3,3}+7x_{4,1}+6x_{4,2}$
	$7+9+5+9+8+2+7+10+3+6\le 66$	$+5x_{4,3}+9x_{5,1}+9x_{5,2}+6x_{5,3}+4x_{6,1}+4x_{6,2}+4x_{6,3}+8x_{7,1}+7x_{7,2}+5x_{7,3}$
	$10+7+7+7+9+4+5+4+1+3\le 59$	$+4x_{8,1}+3x_{8,2}+2x_{8,3}+6x_{9,1}+6x_{9,2}+1x_{9,3}+9x_{10,1}+10x_{10,2}+3x_{10,3}\le 59$

RCPSP تولید شده:

$\{x_{1,1}, x_{2,2}, x_{3,1}, x_{4,3}, x_{5,2}, x_{6,3}, x_{7,1}, x_{8,3}, x_{9,3}, x_{10,3}, x_{11,3}, x_{12,2}\}$

j1234_6 بررسی شدنی بودن RCPSP تولید شده:

$9+2+3+2+6+2+4+5+4+5+4+2 \le 48$

$7+4+8+6+9+8+2+5+4+6+9+5 \le 73$

RCPSP تولید شده:

$\{x_{1,1}, x_{2,1}, x_{3,1}, x_{4,1}, x_{5,1}, x_{6,1}, x_{7,1}, x_{8,3}, x_{9,2}, x_{10,3}, x_{11,3}, x_{12,3}, x_{13,3}, x_{14,2}\}$

j1445_3 بررسی شدنی بودن RCPSP تولید شده:

$4+6+6+9+5+7+9+6+4+5+3+6+1+7 \le 78$

$7+6+5+6+6+9+2+4+4+7+7+2+2+3 \le 72$

RCPSP تولید شده:

$\{x_{1,1}, x_{2,1}, x_{3,1}, x_{4,1}, x_{5,1}, x_{6,1}, x_{7,1}, x_{8,1}, x_{9,1}, x_{10,1}, x_{11,1}, x_{12,1}, x_{13,1}, x_{14,1}, x_{15,1}, x_{16,1}\}$

بررسی شدنی بودن RCPSP تولید شده:

$0+4+0+0+0+10+0+0+5+0+0+0+7+1+4+7 \le 45$

j1623_1

$8+0+7+8+2+0+6+10+0+1+9+6+0+0+0+0 \le 69$

$9x_{1,1}+5x_{1,2}+4x_{1,3}+3x_{2,1}+2x_{2,2}+1x_{2,3}+3x_{3,1}+4x_{3,2}+3x_{3,3}+9x_{4,1}+6x_{4,2}+2x_{4,3}+9x_{5,1}+6x_{5,2}+3x_{5,3}+6x_{6,1}+3x_{6,2}+2x_{6,3}+4x_{7,1}+3x_{7,2}+4x_{7,3}+10x_{8,1}+8x_{8,2}+5x_{8,3}+6x_{9,1}+4x_{9,2}+4x_{9,3}+6x_{10,1}+6x_{10,2}+5x_{10,3}+5x_{11,1}+4x_{11,2}+4x_{11,3}+6x_{12,1}+2x_{12,2}+4x_{12,3} \le 48$

$7x_{1,1}+7x_{1,2}+6x_{1,3}+7x_{2,1}+4x_{2,2}+3x_{2,3}+8x_{3,1}+9x_{3,2}+7x_{3,3}+7x_{4,1}+7x_{4,2}+6x_{4,3}+10x_{5,1}+9x_{5,2}+9x_{5,3}+10x_{6,1}+9x_{6,2}+8x_{6,3}+2x_{7,1}+2x_{7,2}+2x_{7,3}+6x_{8,1}+6x_{8,2}+5x_{8,3}+5x_{9,1}+5x_{9,2}+4x_{9,3}+8x_{10,1}+8x_{10,2}+6x_{10,3}+10x_{11,1}+10x_{11,2}+9x_{11,3}+5x_{12,1}+5x_{12,2}+4x_{12,3} \le 73$

$4x_{1,1}+2x_{1,2}+2x_{1,3}+6x_{2,1}+4x_{2,2}+2x_{2,3}+6x_{3,1}+5x_{3,2}+4x_{3,3}+9x_{4,1}+4x_{4,2}+6x_{4,3}+5x_{5,1}+4x_{5,2}+1x_{5,3}+7x_{6,1}+6x_{6,2}+6x_{6,3}+9x_{7,1}+9x_{7,2}+8x_{7,3}+9x_{8,1}+8x_{8,2}+6x_{8,3}+6x_{9,1}+4x_{9,2}+4x_{9,3}+8x_{10,1}+8x_{10,2}+5x_{10,3}+7x_{11,1}+5x_{11,2}+3x_{11,3}+7x_{12,1}+7x_{12,2}+6x_{12,3}+5x_{13,1}+3x_{13,2}+1x_{13,3}+8x_{14,1}+7x_{14,2}+7x_{14,3} \le 78$

$7x_{1,1}+6x_{1,2}+5x_{1,3}+6x_{2,1}+5x_{2,2}+5x_{2,3}+5x_{3,1}+4x_{3,2}+2x_{3,3}+6x_{4,1}+4x_{4,2}+3x_{4,3}+6x_{5,1}+6x_{5,2}+6x_{5,3}+9x_{6,1}+7x_{6,2}+6x_{6,3}+2x_{7,1}+2x_{7,2}+2x_{7,3}+10x_{8,1}+6x_{8,2}+4x_{8,3}+7x_{9,1}+4x_{9,2}+3x_{9,3}+10x_{10,1}+9x_{10,2}+7x_{10,3}+8x_{11,1}+7x_{11,2}+7x_{11,3}+3x_{12,1}+3x_{12,2}+2x_{12,3}+6x_{13,1}+5x_{13,2}+2x_{13,3}+3x_{14,1}+3x_{14,2}+1x_{14,3} \le 72$

$0x_{1,1}+3x_{1,2}+1x_{1,3}+4x_{2,1}+0x_{2,2}+0x_{2,3}+0x_{3,1}+0x_{3,2}+0x_{3,3}+0x_{4,1}+0x_{4,2}+0x_{4,3}+0x_{5,1}+0x_{5,2}+0x_{5,3}+10x_{6,1}+0x_{6,2}+9x_{6,3}+0x_{7,1}+0x_{7,2}+10x_{7,3}+0x_{8,1}+0x_{8,2}+0x_{8,3}+5x_{9,1}+0x_{9,2}+4x_{9,3}+0x_{10,1}+7x_{10,2}+7x_{10,3}+0x_{11,1}+0x_{11,2}+0x_{11,3}+0x_{12,1}+0x_{12,2}+0x_{12,3}+7x_{13,1}+7x_{13,2}+0x_{13,3}+1x_{14,1}+0x_{14,2}+0x_{14,3}+4x_{15,1}+0x_{15,2}+0x_{15,3}+7x_{16,1}+7x_{16,2}+5x_{16,3} \le 45$

$8x_{1,1}+0x_{1,2}+0x_{1,3}+0x_{2,1}+7x_{2,2}+1x_{2,3}+7x_{3,1}+6x_{3,2}+4x_{3,3}+8x_{4,1}+4x_{4,2}+1x_{4,3}+2x_{5,1}+2x_{5,2}+2x_{5,3}+0x_{6,1}+1x_{6,2}+0x_{6,3}+6x_{7,1}+5x_{7,2}+0x_{7,3}+10x_{8,1}+8x_{8,2}+7x_{8,3}+0x_{9,1}+4x_{9,2}+0x_{9,3}+1x_{10,1}+0x_{10,2}+0x_{10,3}+9x_{11,1}+9x_{11,2}+9x_{11,3}+6x_{12,1}+3x_{12,2}+2x_{12,3}+0x_{13,1}+0x_{13,2}+2x_{13,3}+0x_{14,1}+3x_{14,2}+2x_{14,3}+0x_{15,1}+9x_{15,2}+5x_{15,3}+0x_{16,1}+0x_{16,2}+0x_{16,3} \le 69$

۴-۵. ارزیابی راهکارهای پیشنهاد شده

عملکرد هر یک از پیشنهادات این مقاله در قالب پنج سناریوی طراحی شده با استفاده از ۲۰ نمونه تصادفی از هر یک از دسته‌های J10، J12، J14 و J16 از مجموعه مسایل استاندارد *کولیش* تحلیل شده است. داده‌های بدست آمده با توجه به چهار معیار تعیین شده به‌طور مجزا جمع‌آوری و میانگین نتایج مربوط به هر آزمون محاسبه شده است. سپس با کمک اطلاعات حاصله میزان تاثیرگذاری هر یک از سناریوها با محاسبه درصد اختلاف میانگین حاصل از اجرای آزمون الگوریتم پایه از هر یک از سناریوها: مرتب‌سازی، قاعده هرس۱، قاعده هرس۲ و بالاخره الگوریتم نهایی بدست آمده و در جدول‌های ۲ تا ۵ آورده شده‌اند. در این جدول‌ها ستون اول حاوی نام دسته مورد آزمون و ستون‌های بعدی در هر جدول با توجه به معیار اندازه‌گیری به ترتیب، شامل میانگین تعداد عبارت‌های تضاد یافت شده (جدول شماره ۲)، میانگین طول متوسط هر عبارت تضاد (جدول شماره ۳)، میانگین زمان اجرا (جدول شماره ۴) و میانگین تعداد گره‌های ملاقات شده (جدول شماره ۵) در هر دسته می‌باشد. بدیهی است هر چه میزان بهبود محاسبه شده بیشتر باشد، تاثیر مثبت گام‌های افزوده شده به الگوریتم پایه در جهت بهبود مشکل حافظه و زمان بیشتر است. در ادامه، عملکرد هر یک از راهکارهای پیشنهاد شده را در جهت بهبود الگوریتم پایه با جزییات بیشتر مورد بررسی قرار می‌دهیم.

۱-۵-۴. ارزیابی تعداد عبارت‌های تضاد

درصد اختلاف میانگین تعداد عبارت‌های تضاد خارج شده در آزمون الگوریتم پایه از تعداد عبارت‌های تضاد خارج شده در هر

یک از نمونه‌های استاندارد تحت هر یک از آزمون‌های پیشنهاد شده، در جدول ۲ آورده شده است.

جدول. ۲. درصد بهبود میانگین تعداد عبارت های تضاد

نام دسته	راهکارهای پیشنهادی			
	مرتب سازی	هرس ۱	هرس ۲	الگوریتم نهایی
$J10$	۶۷/۶۷	۶۷/۶۷	۶۷/۶۷	۶۷/۶۷
$J12$	۷۹/۱۵	۷۹/۱۵	۷۹/۱۵	۷۹/۱۵
$J14$	۸۳/۳۹	۸۳/۳۹	۸۳/۳۹	۸۳/۳۹
$J16$	۸۸/۷۸	۸۸/۷۸	۸۸/۷۸	۸۸/۷۸

به عنوان مثال، با نگاهی به دسته J16 جدول ۲ مشاهده می‌شود که هر یک از پیشنهادات این مقاله عملکرد الگوریتم پایه را دست‌کم به میزان تقریبی ۸۸٪ بهبود بخشیده است. استفاده از مرتب‌سازی در همه سناریوها که موجب می‌شود عبارت ورودی به درخت شمارشی در هر چهار سناریو یکسان باشد، باعث گردیده تا هر یک از راهکارهای پیشنهادی، الگوریتم پایه را به یک میزان بهبود ببخشند. اگر چه قواعد هرس۱ و هرس۲ تاثیری در بهبود معیار تعداد عبارت‌های تضاد ندارد اما اثر مثبت آن‌ها برروی معیارهای دیگر در ادامه قابل مشاهده است.

۴-۵-۲. ارزیابی طول متوسط هر عبارت تضاد

درصد اختلاف میانگین طول متوسط عبارت‌های تضاد خارج شده از نمونه‌های استاندارد در سناریوی الگوریتم پایه از میانگین طول متوسط عبارت‌های تضاد خارج شده از نمونه‌های استاندارد تحت هر یک از سناریوهای پیشنهادی در جدول ۳ آورده شده است.

جدول. ۳. درصد بهبود میانگین طول متوسط عبارت‌های تضاد

نام دسته	راهکارهای پیشنهادی			
	مرتب سازی	هرس ۱	هرس ۲	الگوریتم نهایی
$J10$	۱۲/۲۰	۱۲/۲۰	۱۲/۲۰	۱۲/۲۰
$J12$	۱۰/۰۸	۱۰/۰۸	۱۰/۰۸	۱۰/۰۸
$J14$	۱۲/۲۳	۱۲/۲۳	۱۲/۲۳	۱۲/۲۳
$J16$	۱۵/۶۵	۱۵/۶۵	۱۵/۶۵	۱۵/۶۵

داده‌های جدول ۳ نشان می‌هند که پیشنهادات این مقاله، عملکرد الگوریتم پایه را نسبت به این معیار بین ۱۰٪ تا ۱۵٪ بهبود داده‌اند. لازم به ذکر است که در بررسی معیار طول متوسط عبارت‌های تضاد مانند معیار تعداد عبارت‌های تضاد به علت ساختار یکسان عبارت ورودی در هر چهار سناریوی مرتب‌سازی، قاعده هرس۱، قاعده هرس۲ و حالت الگوریتم نهایی، میزان بهبود میانگین طول متوسط عبارت‌های تضاد یکسان است. به‌عبارت دیگر عمل مرتب‌سازی در کاهش طول متوسط عبارت‌های تضاد بیشترین تاثیر را داشته است.

۴-۵-۳. ارزیابی زمان اجرا

چون سخت‌افزار و نرم‌افزار مورد استفاده برای اجرای الگوریتم، تاثیر مستقیم بر زمان اجرا دارد. لذا لازم است تا اجرای پنج سناریو به ازای هر نمونه تنها بر روی یک کامپیوتر انجام گیرد. میانگین زمان اجرای هر آزمون بر روی دسته‌های J10 تا J16 بر حسب ثانیه محاسبه شده است. جدول ۴ میزان دقیق اثرگذاری هر یک از سناریوهای پیشنهادی را نشان می‌دهد.

جدول. ۴. درصد بهبود میانگین زمان اجرا

نام دسته	راهکارهای پیشنهادی			
	مرتب سازی	هرس ۱	هرس ۲	الگوریتم نهایی
$J10$	۸۴/۹۴	۸۵/۴۴	۸۵/۴۴	۸۶/۰۸
$J12$	۹۴/۵۹	۹۴/۷۰	۹۴/۸۲	۹۴/۹۲
$J14$	۹۵/۸۰	۹۵/۷۸	۹۵/۶۹	۹۵/۸۲
$J16$	۹۸/۸۱	۹۸/۸۰	۹۸/۸۲	۹۸/۸۱

در جدول ۴ برای مثال مشاهده می‌شود که پیشنهادات این مقاله در مقایسه با الگوریتم پایه توانسته‌اند میانگین زمان لازم برای اجرای دسته J16 را بالغ بر ۹۸٪ بهبود دهند.

۴-۵-۴. ارزیابی تعداد گره‌های ملاقات شده

جدول ۵ نتایج عملکرد سناریوهای پیشنهادی این مقاله را در مورد معیار تعداد گره های ملاقات شده نشان می دهد.

جدول. ۵. درصد بهبود میانگین تعداد گره‌های ملاقات شده

نام دسته	راهکارهای پیشنهادی			
	مرتب سازی	هرس ۱	هرس ۲	الگوریتم نهایی
$J10$	۷۴/۶۴	۷۹/۲۰	۸۲/۶۵	۸۴/۴۹
$J12$	۸۴/۷۶	۸۷/۷۱	۸۹/۱۹	۹۰/۶۲
$J14$	۸۶/۹۷	۸۹/۹۱	۹۱/۱۹	۹۲/۳۷
$J16$	۹۱/۷۷	۹۳/۵۶	۹۴/۰۱	۹۵/۰۲

با توجه به جدول ۵ ملاحظه می‌شود که مثلا در دسته J16، مرتب‌سازی بالغ بر ۹۱٪، قاعده هرس۱ بالغ بر ۹۳٪، قاعده هرس۲ بالغ بر ۹۴٪ و بالاخره الگوریتم نهایی بالغ بر ۹۵٪ گره کمتری نسبت به الگوریتم پایه ملاقات کرده‌اند.

۴-۵-۵. مقیاس‌پذیری

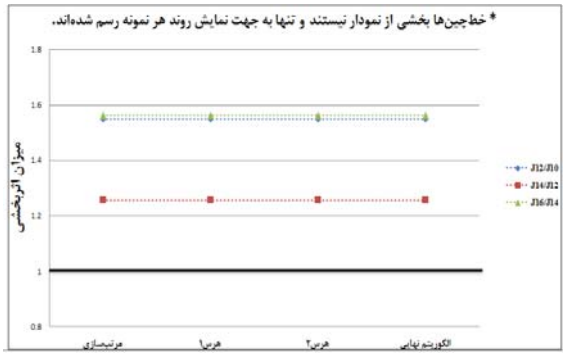
نتایج بدست آمده از ارزیابی‌ها حاکی از آن است که پیشنهادات این مقاله توانسته‌اند در حل مشکل حافظه و زمان الگوریتم پیمایش درخت شمارشی به میزان قابل توجهی موثر باشند. در دنیای واقعی تعداد فعالیت‌های یک پروژه بیش از آن چیزی است که در این

مقاله مورد بررسی قرار گرفته است. بنابراین مساله‌ای که در اینجا مطرح می‌شود، این است که آیا این روند بهسازی هر یک از سناریوهای پیشنهادی نسبت به الگوریتم پایه می‌تواند به دسته مسایل بزرگتر نیز تعمیم داده شود. برای پاسخگویی به این سوال ویژگی مقیاس‌پذیری نسبت الگوریتم پایه به پیشنهادات، مورد بررسی قرار گرفته است، چرا که آزمودن نمونه‌هایی با ابعاد بزرگ به علت حجم بالای محاسبات نیاز به صرف زمان زیاد و امکانات پیشرفته‌تر از آنچه در دسترس است، دارد.

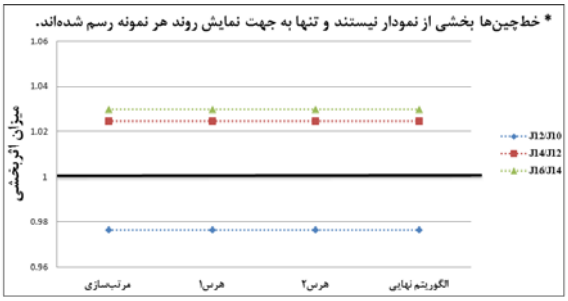
مقیاس‌پذیری، ویژگی مطلوبی از یک سیستم، شبکه و یا فرایند است که به توانایی آن برای پاسخگویی به افزایش میزان بار کاری دلالت می‌کند و یا میزان آمادگی سیستم را برای افزایش بار کاری نشان می‌دهد [16]. سامانه‌ای که کارایی آن با افزایش ظرفیت افزایش می‌یابد مقیاس‌پذیر خوانده می‌شود. برای بررسی مقیاس‌پذیری اثربخشی محاسبه می‌شود. چنانچه اثربخشی بزرگتر یا مساوی یک باشد سیستم مقیاس‌پذیر است و در غیر این صورت مقیاس‌پذیر نمی‌باشد. در این بخش برای بررسی مقیاس‌پذیری، میزان بهبود مرتب‌سازی، قاعده هرس۱، قاعده هرس۲ و الگوریتم نهایی نسبت به الگوریتم پایه بر مبنای هر چهار معیار، مورد بررسی قرار گرفته است. اثربخشی به شکل رابطه (۱۳) تعریف می‌شود.

$$\text{اثربخشی} = \frac{\text{میانگین میزان بهبود در مساله با اندازه بزرگ}}{\text{میانگین میزان بهبود در مساله با اندازه کوچک}} \qquad (۱۳)$$

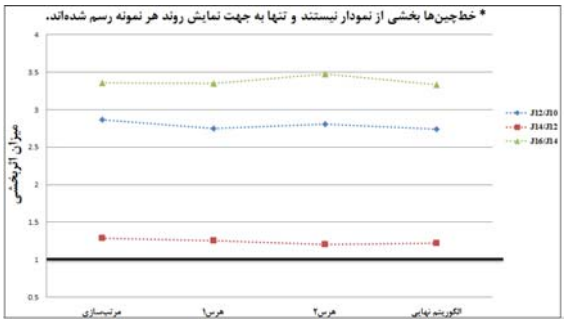
برای بررسی مقیاس‌پذیری، اثربخشی J12 نسبت به J10، J14 نسبت به J12، J16 نسبت به J14 برای هر پیشنهاد محاسبه شده و نتایج مربوط به هر معیار در شکل‌های ۷ تا ۱۰ نشان داده شده است. در این شکل‌ها، محور افقی نشانگر پیشنهاد و محور عمودی نشانگر میزان اثربخشی است.



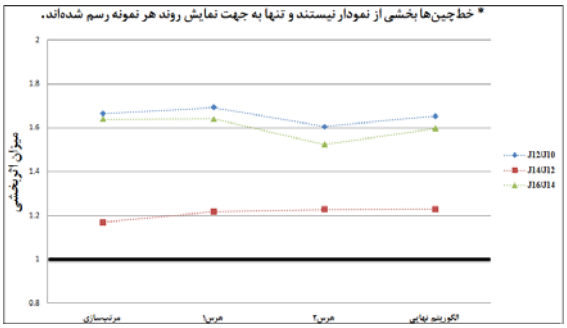
شکل. ۶. میزان اثربخشی بر مبنای معیار میانگین تعداد عبارت‌های تضاد



شکل. ۷. میزان اثربخشی بر مبنای معیار طول متوسط عبارت‌های تضاد



شکل. ۸. میزان اثربخشی بر مبنای معیار زمان اجر



شکل. ۹. میزان اثربخشی بر مبنای تعداد گره‌های ملاقات شده

با توجه به شکل‌های ۷، ۹ و ۱۰ مشاهده می‌شود که میزان اثربخشی معیارهای ذیربط کاملاً بالاتر از ۱ قرار گرفته است. این موضوع حاکی از آن است که میزان بهبود سناریوهای پیشنهادی نسبت به الگوریتم پایه به ترتیب از جهت معیار میانگین تعداد عبارت‌های تضاد، میانگین طول زمان اجرا و میانگین تعداد گره ملاقات شده مقیاس‌پذیر است و با بزرگتر شدن مساله، سربار پردازشی ناشی از اضافه کردن پیشنهاد مربوطه آنقدر ناچیز بوده که مانع از بهبود روند حل نشده و روند بهبود ادامه یافته است. اما در شکل ۸ مشاهده می‌شود که نمودار مربوط به نسبت‌های دسته

مسایل J14 به J12 و J16 به J14 کاملاً بالاتر از یک و نمودار مربوط به نسبت دسته مسایل J12 به J10 کاملا پایین‌تر از یک واقع شده است است. این موضوع حاکی از آن است که سربار پردازشی ناشی از اضافه کردن پیشنهادات در ابتدا با بزرگتر شدن مساله مانع بهبود روند حل شده است اما با ادامه‌ی روند بزرگتر شدن مساله، مشاهده می‌شود که اثر این سربار پردازشی، بهبود الگوریتم از جهت معیار طول متوسط عبارت‌های تضاد و مقیاس‌پذیر شدن آن را به دنبال داشته است.

۵. نتایج و پیشنهادات

در این مقاله یک روش فراابتکاری برای حل مساله زمان‌بندی پروژه با منابع محدود چندحالته معرفی شده که از یک درخت شمارشی و یک حل‌کننده مساله صدق‌پذیری برای تبدیل این مساله به مساله زمان‌بندی پروژه با منابع محدود تک‌حالته استفاده می‌کند. این روش حل در برخی نمونه‌ها به زمان محاسباتی و حافظه بالایی بویژه در پیمایش درخت شمارشی نیاز دارد. برای بهبود عملکرد این روش، سه پیشنهاد ارایه گردید و مورد آزمون قرار گرفت. نتایج حاکی از آن است که اضافه کردن پیشنهادات مرتب‌سازی، قاعده هرس۱ و قاعده هرس۲ به الگوریتم پایه درخت شمارشی موجب افزایش کارایی این روش می‌شود و مشکل زمان و حافظه را به میزان قابل توجهی بهبود می‌دهد. با بررسی ویژگی مقیاس‌پذیری مشخص شد که روند بهبود روش با اضافه کردن پیشنهادات، علی رغم بزرگتر شدن نمونه‌ها ادامه می‌یابد.

روش فراابتکاری معرفی شده طی دو گام مجزا مساله زمان‌بندی پروژه با منابع محدود چندحالته را حل می‌کند. در گام اول فارغ از ماهیت تابع هدف و محدودیت‌ها، مساله‌های متفاوت زمان‌بندی پروژه با منابع محدود تک‌حالتی می‌توانند تولید شوند که برتری هیچ یک نسبت به دیگری از منظر حل‌پذیری قابل تشخیص نیست. بنابراین پیشنهاد می شود تا در مطالعات آتی حل گام اول با نگاهی به تابع هدف و محدودیت‌های گام دوم اجرا شود تا شرایط برای رسیدن به جواب بهینه هموارتر شود.

موضوع دیگری که می‌تواند در تحقیقات آتی مورد مطالعه قرار گیرد، درنظر گرفتن یک محدودیت جدید برای منابع در مدل مساله زمان‌بندی پروژه با منابع محدود است. زیرا تقسیم منابع به دو دسته تجدیدپذیر و تجدیدناپذیر اگرچه از دید مصرف‌پذیری و بازه زمانی تعیین‌شده، تقسیم‌بندی معقولی به‌نظر می‌رسد، اما آنچه در واقعیت اتفاق می‌افتد این است که به مرور زمان منابع تجدیدپذیر مانند ماشین‌آلات، نیروی انسانی و غیره کارایی اولیه خود را از دست می‌دهند و گاه حتی همانند یک منبع تجدیدناپذیر به پایان می‌رسند.

مراجع

[۱] مهدی‌زاده، اسماعیل.، معلمیان، علی و حاجی‌پور، وحید.، «مساله زمان‌بندی پروژه با منابع محدود با هدف حداکثر کردن ارزش خالص فعلی»، نشریه بین‌المللی صنایع و مدیریت تولید، جلد ۲۵، شماره ۴، صفحه ۴۰۱-۳۹۰، ۱۳۹۳.

[2] Herroelen W., Demeulemeester E., & De Reyck, B. " A classification scheme for project scheduling", Springer US ,1999, PP. 1-26.

[3] Słowinski R., "Multiobjective network scheduling with efficient use of renewable and nonrenewable resources" ,European Journal of Operational Research, Vol. 7, No. 3, 1981, PP. 265-273.

[4] Kolisch R., "Project scheduling under resource constraints: efficient heuristics for several problem classes", Springer Verlag, 1995.

[5] Sprecher A., "Resource Constrained Project Scheduling: Exact Methods for the Multi-mode Case", Lecture Notes in Economics and Mathematical Systems, Springer, Berlin, 1994.

[6] Hartmann S., & Drexl A., "Project scheduling with multiple modes: A comparison of exact algorithms", Networks, Vol. 32, No. 4, 1998, PP. 283-297.

[7] Zhu G., Bard J.F., & Yu G., "A branch-and-cut procedure for the multimode resource-constrained project-scheduling problem", INFORMS Journal on Computing, Vol. 18, No. 3, 2006, PP. 377-390.

[8] Ozdamar L., "A genetic algorithm approach to a general category project scheduling proble." Systems Man and

Cybernetics Part C: Applications and Reviews, IEEE Transactions , Vol. 29, No. 1, 1999, PP. 44-59.

[9] Lova A., Tormos P., & Barber F., "Multi-mode resource constrained project scheduling: Scheduling schemes, priority rules and mode selection rules", Inteligencia artificial, Vol. 30, No. 10, 2006, PP. 69-86.

[10] Van Peteghem V., Vanhoucke M., "An artificial immune system for the multi-mode resource-constrained project scheduling problem", In Evolutionary Computation in Combinatorial Optimization, Springer Berlin Heidelberg, 2009, PP. 85-96.

[11] Yoosefzadeh, H. R., & Tareghian, H. R. " Hybrid solution method for resource-constrained project scheduling problem using a new schedule generator", The International Journal of Advanced Manufacturing Technology, Vol. 66, No. 5-8, 2013, PP. 1171-1180.

[12] Beşikci U., Bilge Ü., & Ulusoy G., "Multi-mode resource constrained multi-project scheduling and resource portfolio proble", European Journal of Operational Research, Vol. 240, No. 1, 2015, PP. 22-31.

[13] Coelho J,. Vanhoucke M., "Multi-mode resource-constrained project scheduling using RCPSP and SAT solvers", European Journal of Operational Research, Vol. 213, No. 2, 2011, PP. 73-82.

[14] Alves R., Alvelos F., & Sousa S. D., " Resource Constrained Project Scheduling with General Precedence Relations Optimized with SAT", In Progress in Artificial Intelligence, Springer Berlin Heidelberg, 2013, PP. 199-210.

[15] Kolisch R,. & Drexl A., "Local search for nonpreemptive multi-mode", IIE transactions, Vol. 29, No. 11, 1997, PP. 987-999.

[16] Bondi A. B., "Characteristics of scalability and the irimpact on performance", In Proceedings of the 2nd international workshop on Software and performance, Sep. 2000, PP. 195-203.