



## حل مساله کوله پشتی چندبعدی با استفاده از اتوماتاهای یادگیر

سمیرا نوفرستی

عضو هیات علمی دانشگاه سیستان و بلوچستان

دانشکده مهندسی شهید نیکبخت، گروه فناوری اطلاعات

[snoferesti@ece.usb.ac.ir](mailto:snoferesti@ece.usb.ac.ir)

محمد رضا میبدی

عضو هیات علمی دانشگاه صنعتی امیرکبیر

دانشکده مهندسی کامپیوتر و فناوری اطلاعات

[mmevodi@aut.ac.ir](mailto:mmevodi@aut.ac.ir)

**چکیده:** در این مقاله یک الگوریتم تکرارشونده مبتنی بر اتوماتاهای یادگیر برای حل مساله کوله پشتی چندبعدی پیشنهاد می شود. در این الگوریتم، مساله کوله پشتی با یک گراف کامل مدل می شود که هر گره از گراف متناظر با یکی از کالاهاست. هر گره از گراف به یک اتوماتای یادگیر مجهز است که انتخاب یا عدم انتخاب کالای متناظر با گره برای قرار گرفتن در کوله پشتی را مشخص می کند. نتایج شبیه سازی ها نشان داده است که الگوریتم پیشنهادی در مقایسه با الگوریتمهای موجود از کارایی بالاتری برخوردار است. نتایج شبیه سازیها همچنین نشان داده است که الگوریتم پیشنهادی برای مسائل با اندازه های بزرگ دارای سرعت همگرایی بالایی می باشد.

**واژه های کلیدی:** مساله کوله پشتی چندبعدی، اتوماتاهای یادگیر، مسایل مشکل

### ۱- مقدمه

مساله کوله پشتی چندبعدی<sup>۱</sup> یکی از مسایل بهینه سازی ترکیبی است که کاربردهای فراوانی از قبیل تخصیص منابع، برنامه ریزی بودجه، تخصیص سهام و بارگیری محموله ها دارد. مساله کوله پشتی چندبعدی شامل  $m$  کوله پشتی با ظرفیتهای  $b_1, b_2, \dots, b_m$  و  $n$  کالا است. کالای  $j$ ام دارای ارزش  $p_j$  می باشد و وزن  $w_{ji}$  از کوله پشتی  $m_j$  را اشغال می کند. هدف پر کردن کوله پشتی ها با زیرمجموعه ای از کالاها است به نحوی که بیشترین سود حاصل شود و مجموع وزن کالاهای یک کوله پشتی از ظرفیت آن تجاوز نکند. یک کالا یا در همه کوله پشتی ها قرار می گیرد یا برای هیچیک از کوله پشتی ها انتخاب نمی شود. به طور دقیقتر می توان مساله کوله پشتی را به صورت زیر تعریف کرد:

$$\begin{aligned} \max \quad & \sum_{j=1}^n p_j x_j \\ \text{subject to} \quad & C_i : \sum_{j=1}^n w_{ij} x_j \leq b_i, \quad \forall i \in 1..m \\ & x_j \in \{0,1\}, \quad \forall j \in 1..n \end{aligned} \quad (1)$$

<sup>۱</sup> Multiple knapsack problem

که  $x_j$  متغیر تصمیم گیری متناظر با کالای  $j$ ام است. در صورتی که کالای  $j$ ام انتخاب شود،  $x_j$  مقدار ۱ و در غیر این صورت مقدار صفر را اختیار می کند. متغیرهای  $w_{ij}$  و  $b_i$  نمی توانند مقادیر منفی اختیار کنند.

به دلیل اهمیت مساله کوله پشتی تاکنون الگوریتمهای متعددی برای حل آن گزارش شده است. این الگوریتمها را می توان به دو گروه کلی تقسیم کرد: الگوریتمهای دقیق و الگوریتمهای تقریبی. با توجه به اینکه مساله کوله پشتی یک مساله NP\_Complete می باشد الگوریتمهای دقیق که معمولاً از روشهای برش وانشعاب<sup>۲</sup> یا روشهای ترکیبی با برنامه نویسی پویا<sup>۳</sup> استفاده می کنند [۱،۲،۳،۴] در بدترین حالت دارای پیچیدگی نمایی هستند و برای استفاده در کاربردهای عملی مناسب نمی باشند. به همین دلیل الگوریتمهای تقریبی متعددی برای حل مساله کوله پشتی گزارش شده است. بسیاری از تحقیقات انجام شده به حل مساله کوله پشتی تک بعدی<sup>۴</sup> ( $m=1$ ) پرداخته اند [۵،۶]. تلاشهای زیادی نیز در جهت حل تقریبی مساله کوله پشتی چندبعدی انجام گرفته است. از جمله الگوریتمهای تقریبی برای حل مساله کوله پشتی چندبعدی، الگوریتمهای تکرارشونده<sup>۵</sup> هستند. در این روشها رسیدن به یک پاسخ بهینه تضمین نمی شود اما در اغلب موارد جوابهای تقریبی قابل قبولی تولید می کنند. از جمله الگوریتمهای تکرارشونده می توان به الگوریتم ژنتیکی [۷،۸،۹] و الگوریتم کلونی مورچه ها [۱۰،۱۱] اشاره کرد.

در این مقاله یک الگوریتم تکرارشونده مبتنی بر اتوماتاهای یادگیر برای حل مساله کوله پشتی چندبعدی پیشنهاد می شود. در این الگوریتم مساله کوله پشتی با یک گراف کامل مدل می شود که هر گره از گراف متناظر با یکی از کالاهاست. هر گره از گراف به یک اتوماتای یادگیر با دو عمل مجهز است که انتخاب یا عدم انتخاب کالای متناظر برای قرار گرفتن در کوله پشتی را مشخص می کند. در هر تکرار از الگوریتم

<sup>۲</sup> Branch and cut

<sup>۳</sup> Dynamic programming

<sup>۴</sup> Uni-dimensional

<sup>۵</sup> Iterative

## ۲-۱ اتوماتای یادگیر با ساختار متغیر

اتوماتای یادگیر با ساختار متغیر توسط  $\delta$  تایی  $LA \equiv \{a, b, p, T, c\}$  نشان داده می‌شود که  $a \equiv \{a_1, a_2, \dots, a_r\}$  مجموعه عملهای اتوماتای یادگیر،  $b \equiv \{b_1, b_2, \dots, b_m\}$  مجموعه ورودی‌های اتوماتای یادگیر،  $P \equiv \{p_1, p_2, \dots, p_r\}$  بردار احتمال انتخاب عملها،  $T \equiv p(n+1) = T[a(n), b(n), p(n)]$  الگوریتم یادگیری و  $c \equiv \{c_1, c_2, \dots, c_r\}$  احتمال جریمه شدن هر عمل می‌باشند. اگر در اتوماتای یادگیر عمل  $a_i$  در مرحله  $n$  انتخاب شود و پاسخ مطلوب از محیط دریافت نماید، احتمال  $P_i(n)$  افزایش یافته و سایر احتمالات کاهش می‌یابند و برای پاسخ نامطلوب احتمال  $P_i(n)$  کاهش یافته و سایر احتمالات افزایش می‌یابند. تغییرات به گونه‌ای صورت می‌گیرد تا حاصل جمع  $P_i(n)$  ها همواره ثابت و مساوی یک باقی بماند.

الف- پاسخ مطلوب

$$\begin{aligned} p_i(n+1) &= p_i(n) + a[1 - p_i(n)] \\ p_j(n+1) &= (1-b)p_j(n) \quad \forall j \quad j \neq i \end{aligned} \quad (2)$$

ب- پاسخ نامطلوب

$$\begin{aligned} p_i(n+1) &= (1-b)p_i(n) \\ p_j(n+1) &= \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j \quad j \neq i \end{aligned} \quad (3)$$

در روابط فوق،  $a$  پارامتر پاداش و  $b$  پارامتر جریمه می‌باشد. با توجه به مقادیر  $a$  و  $b$  سه حالت مختلف را می‌توان در نظر گرفت. زمانی که  $a$  و  $b$  با هم برابر باشند، الگوریتم را  $L_{RP}$  می‌نامند. زمانی که  $a$  خیلی کوچکتر باشد، الگوریتم را  $L_{REP}$  و زمانی که  $b$  مساوی صفر باشد، الگوریتم را  $L_{RI}$  می‌نامند. برای مطالعه بیشتر در رابطه با اتوماتاهای یادگیر می‌توان به [۱۲، ۱۳] مراجعه نمود.

## ۳- حل مساله کوله‌پشتی توسط اتوماتاهای یادگیر

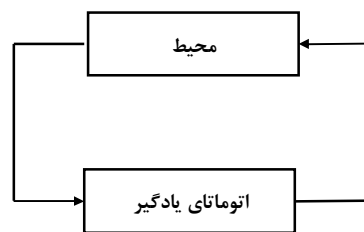
در الگوریتم پیشنهادی که آنرا MKPLA می‌نامیم، مساله با یک گراف کامل مدل می‌شود که هر گره از گراف متناظر با یک کالا در مساله کوله‌پشتی است. هر گره از گراف به یک اتوماتای یادگیر با دو عمل انتخاب کالا برای قرار گرفتن در کوله‌پشتی‌ها و عدم انتخاب کالا مجهز است.

برای فعالسازی اتوماتاهای یادگیر از عاملها استفاده می‌شود. در هر تکرار از الگوریتم تعدادی عامل وجود دارد که هر عامل یک راه‌حل را می‌سازد. در ابتدا یک عامل به صورت تصادفی بر روی یکی از گره‌های گراف قرار می‌گیرد و باعث فعالسازی اتوماتای یادگیر آن گره می‌شود. هرگاه یک اتوماتای یادگیر فعال می‌شود طبق بردار احتمالات انتخاب اعمال خود یکی از دو عمل خود را انتخاب می‌کند. در ابتدای الگوریتم احتمال انتخاب عمل حضور کالا در کوله‌پشتی‌ها برابر ۱ در نظر گرفته

تعدادی عامل بر روی گره‌های گراف قرار داده می‌شوند که وظیفه فعالسازی اتوماتاهای یادگیر را برعهده دارند. هر عامل منجر به یک راه‌حل می‌شود. با توجه به ارزش راه‌حل بدست آمده از هر تکرار، بردار احتمالات اتوماتاهای یادگیر بروز می‌شود. این روند به دفعات تکرار می‌گردد و در خاتمه الگوریتم، بهترین راه‌حل بدست آمده به عنوان جواب نهایی انتخاب می‌شود. نتایج شبیه‌سازی‌های انجام گرفته کارایی الگوریتم پیشنهادی را در مقایسه با دیگر الگوریتمهای گزارش شده نشان می‌دهد. به علاوه برای مسائل با اندازه بزرگ الگوریتم پیشنهادی از سرعت همگرایی بالایی برخوردار است. ادامه مقاله بدین صورت سازماندهی شده است. در ابتدا در بخش ۲ اتوماتاهای یادگیر به صورت اجمالی معرفی می‌گردد. در بخش ۳ الگوریتم پیشنهادی برای حل مساله کوله‌پشتی ارائه می‌شود. در بخش ۴ نتایج شبیه‌سازی‌های انجام گرفته ارائه می‌شود و بخش پایانی مقاله نتیجه‌گیری می‌باشد.

## ۲- اتوماتاهای یادگیر

اتوماتای یادگیر [۱۲] یک مدل انتزاعی است که می‌تواند تعداد محدودی عمل را انجام دهد. هر عمل انتخاب شده توسط محیطی تصادفی ارزیابی می‌گردد و پاسخی به اتوماتای یادگیر داده می‌شود. اتوماتا از این پاسخ استفاده نموده و عمل خود را برای مرحله بعد انتخاب می‌کند. محیط تصادفی را می‌توان با سه‌تایی  $E \equiv \{a, b, c\}$  تعریف نمود که  $a \equiv \{a_1, a_2, \dots, a_r\}$  مجموعه ورودی‌ها،  $b \equiv \{b_1, b_2, \dots, b_m\}$  مجموعه خروجی‌ها و  $c \equiv \{c_1, c_2, \dots, c_r\}$  مجموعه احتمالات جریمه شدن می‌باشند. هرگاه  $b_i$  دو مقداری باشد  $b_i = 1$  به عنوان جریمه و  $b_i = 0$  به عنوان پاداش در نظر گرفته می‌شود.  $c_i$  احتمال اینکه عمل  $a_i$  نتیجه نامطلوب داشته باشد، می‌باشد. در محیط پایدار مقادیر  $c_i$  بدون تغییر باقی می‌مانند، حال آنکه در محیط ناپایدار این مقادیر در طی زمان تغییر می‌کنند. شکل ۱ ارتباط بین اتوماتای یادگیر و محیط را نشان می‌دهد.



شکل (۱): ارتباط بین اتوماتای یادگیر و محیط

اتوماتاهای یادگیر به دو گروه اتوماتای یادگیر با ساختار ثابت و اتوماتای یادگیر با ساختار متغیر تقسیم می‌شوند. در ادامه به شرح مختصری درباره اتوماتاهای یادگیر با ساختار متغیر که در این مقاله استفاده شده است می‌پردازیم.

اگر ارزش بهترین راه حل تکرار فعلی از ارزش بهترین راه حل بدست آمده تاکنون کمتر باشد اعمال انتخاب شده توسط اتوماتاهای یادگیر گرهای متناظر با کالاهای شرکت کننده در بهترین راه حل پاداش و اعمال انتخاب شده توسط اتوماتاهای یادگیر دیگر گرهای جرمه می شوند. برای جرمه عمل انتخابی از فرمول زیر استفاده می شود.

$$p(t+1) = p(t) - g.b.p(t) \quad (۶)$$

که  $g = \left| \frac{1}{f(c) - c_{\max}} + 1 \right|$  و  $b$  پارامتر جرمه نامیده می شود. کل این فرآیند به صورت تکراری و به دفعات انجام می شود تا زمانی که راه حل بهینه حاصل شود یا شرط ماکزیم تعداد تکرارهای الگوریتم برقرار شود. شبه کد الگوریتم MKPLA در شکل ۳ نشان داده شده است. هر مقاله باید شامل این بخشهای اصلی باشد: چکیده، کلمات کلیدی، مقدمه، مطالب اصلی، نتیجه، و مراجع. سایر بخشها مثل سپاسگزاری، ضمایم، و زیرنویسها اختیاری است. این بخشها باید در آخر مقاله و قبل از مراجع قرار گیرند، بجز بخش زیرنویسها که پس از مراجع آورده می شود.

#### ۴- نتایج شبیه سازی های انجام شده

برای ارزیابی کارایی الگوریتم MKPLA از مجموعه تست پیشنهادی Chu و Beasley [۱۵، ۱۶] که در OR-library [۱۷] موجود است، استفاده شده است. نتایج حاصل از الگوریتم پیشنهادی با چهار الگوریتم دیگر که در [۱۱] گزارش شده اند، مقایسه شده است. جزئیات این الگوریتمها در مراجع [۱۱، ۱۶، ۱۸ و ۱۹] ارائه شده است

در آزمایشات انجام گرفته مقدار پارامتر تصمیم گیری  $q$  برابر  $0.3$ ، مقدار پارامتر پاداش  $a$  برابر  $1$ ، مقدار پارامتر  $\alpha$  برابر  $1$  و مقدار پارامتر  $\beta$  برابر  $3$  در نظر گرفته شده است. به علاوه تعداد عاملها در هر تکرار برابر  $30$  انتخاب شده است. مقدار پارامتر جرمه  $b$  در ابتدا  $1$  در نظر گرفته شده است اما اگر بهترین نتیجه حاصل شده در  $50$  تکرار مجدداً حاصل شد مقدار  $b$  به میزان  $0.3$  کاهش می یابد. جدول ۱ نتایج حاصل برای  $20$  نمونه تست با  $100$  کالا و  $5$  کوله پشتی را نشان می دهد. ستون اول جدول شماره مساله در مجموعه تست، ستون دوم جدول بهترین جواب شناخته شده برای مساله که توسط الگوریتم ژنتیک حاصل شده است [15] و ستونهای بعدی نتایج حاصل از اجرای الگوریتمهای مذکور را نشان می دهد. نتایج حاصل از الگوریتم MKPLA شامل میانگین  $20$  بار اجرای الگوریتم با  $1000$  تکرار در هر اجرا و متوسط تعداد تکرارهای لازم برای رسیدن به بهترین جواب ( $C^*$ ) می باشد. همانطور که از جدول ۱ مشخص است الگوریتم پیشنهادی جوابهای بهتری نسبت به الگوریتمهای گزارش شده تولید می کند.

شده است تا کلیه کالاهای شناس انتخاب را پیدا کنند. پس از فعال شدن یک اتوماتای یادگیر اگر عمل انتخابی آن اتوماتا، انتخاب کالا برای قرار گرفتن در کوله پشتیها باشد، آن کالا را در لیست کالاهای انتخابی قرار می دهیم. سپس عامل از بین گرهای که تاکنون پیمایش نشده اند یک گر را انتخاب می کند. عامل  $k$  گر بعدی را بر اساس رابطه ۴ انتخاب می کند.

$$p(j) = \frac{h_j}{\sum_{i \in \text{validset}} h_i}, \quad h_j = \frac{p_j}{\sum_{i=1}^m \frac{w_{ij}}{S_i}} \quad (۴)$$

که  $\text{validset}$  مجموعه گرهای است که تاکنون توسط عامل  $k$  پیمایش نشده اند و انتخاب آنها باعث انحراف از شرط (۱) نمی شود و  $S_i$  ظرفیت باقی مانده از کوله پشتی  $i$  تا این مرحله می باشد. در این رابطه عامل گرهای را انتخاب می کند که نسبت سود کالای متناظر با آن به مجموع وزن اشغال کننده از ظرفیت باقی مانده کوله پشتیها ماکزیم باشد. در رابطه فوق عامل  $k$  با احتمال  $q$  گر دارای بیشترین احتمال را انتخاب می کند و با احتمال  $1-q$  گر بعدی را با احتمال متناظر با آن گر به صورت تصادفی انتخاب می کند. هر گر تنها یکبار توسط هر عامل پیمایش می شود. هرگاه مجموعه  $\text{validset}$  یک عامل تهی شد کار آن عامل به پایان می رسد و راه حل ایجاد شده توسط آن عامل طبق الگوریتم شکل ۲ بهبود می یابد. برای بهبود راه حل طبق [۱۴] مقادیر LP برای کالاهای محاسبه می شود. سپس کالاهای به ترتیب نزولی مقادیر LP بررسی می شوند. اگر بتوان کالایی را بدون انحراف از شرط (۱) برگزید، آن کالا به لیست کالاهای انتخابی کوله پشتیها اضافه می شود.

```

Procedure improve (  $\hat{x}$  )
    sort  $\hat{x}$  as  $x^{LP[j]} \geq x^{LP[j+1]}$ 
    for  $j = 1$  to  $n$  do
        if  $x[j] = 0$  then  $x[j] = 1$ ;
    if any  $C_i$  is violated then  $x[j] = 0$ ;

```

#### شکل (۲): الگوریتم بهبود راه حل ایجاد شده توسط یک عامل

در هر تکرار از الگوریتم، از ارزش بهترین راه حل ایجاد شده توسط عاملها، برای ارزیابی عمل انتخابی اتوماتاهای یادگیر استفاده می شود. اگر ارزش کالاهای انتخابی بهترین عامل از ارزش بهترین لیست کالاهای بدست آمده تا این مرحله بیشتر باشد، اعمال انتخاب شده توسط اتوماتاهای یادگیر بهترین عامل تکرار فعلی از طریق افزایش احتمال انتخاب آنها پاداش داده می شود. فرمول یادگیری برای پاداش عمل انتخابی بصورت زیر می باشد:

$$p(t+1) = p(t) + q.a.(1 - p(t)) \quad (۵)$$

که  $q = \left| \frac{f(c) - c_{\max}}{f(c)} \right|$  و  $f(c)$  مجموع ارزش کالاهای انتخابی است که در تکرار  $t$ ام تولید شده و  $C_{\max}$  ارزش بهترین راه حل بدست آمده تا این مرحله است.  $a$  پارامتر پاداش نامیده می شود.

**سیزدهمین کنفرانس ملی انجمن کامپیوتر ایران**  
جزیره کیش، خلیج فارس، ایران ۱۹ الی ۲۱ اسفند ۱۳۸۶

**Algorithm MKPLA**

```

construct a complete graph that associates a node to each object;
equip each node with an learning automaton with tow actions {1,2};
set probability of selecting action 1 to 1;
while not maximum number of cycles reached do
  for each agent  $k$ , construct a solution  $Sl_k$  as follows:
    Randomly choose a first object  $O_i$  and active  $LA_i$  in node  $p_i$ ;
    action = the action chosen by  $LA_i$  in node  $p_i$ 
    if action=1 then  $Sl_k = Sl_k \cup \{O_i\}$ ; //select object
    validset= $\{O_i \in \{1,2,\dots,n\} / O_i \notin Sl_k \text{ \& can be selected without violating resource constraint}\}$ 
    while validset $\neq \emptyset$  do
      Rand= a random variable  $\in [0,1]$ 
      if rand $\leq q$  then choose an object $\in$  validset with maximum  $p(i)$ 
      else choose an object $\in$ validset with probability  $ps(i)$ 
      action = the action chosen by  $LA_i$  in node  $p_i$ 
      if action=1 then  $Sl_k = Sl_k \cup \{O_i\}$ ; //select object
      remove from validset every object that violates some resource constraints.
    end while
    improve  $Sl_k$ 
  end for
  update probability vectors of learning automata
end while

```

شکل (۳): الگوریتم MKPLA برای حل مساله کوله پشتی چندبعدی

جدول (۱): مقایسه نتایج الگوریتم MKPLA با دیگر الگوریتمهای گزارش شده برای ۱۰۰ کالا و ۵ کوله پشتی

شماره مساله	بهترین جواب شناخته شده	L. & M. (میانگین)	FIDANOVA (بهترین)	ALAYA میانگین (C*)	MKPLA (میانگین) (C*)
۰	۲۴۳۸۱	۲۴۳۳۱	۲۳۹۸۴	۲۴۳۴۲	۲۴۳۶۴.۰۵
۱	۲۴۲۷۴	۲۴۲۴۵	۲۴۱۴۵	۲۴۲۴۷	۲۴۲۷۰
۲	۲۳۵۵۱	۲۳۵۲۷	۲۳۵۲۳	۲۳۵۲۹	۲۳۵۳۷.۲۵
۳	۲۳۵۳۴	۲۳۴۶۳	۲۲۸۷۴	۲۳۴۶۲	۲۳۴۹۵.۳
۴	۲۳۹۹۱	۲۳۴۶۹	۲۳۷۵۱	۲۳۹۴۶	۲۳۹۵۹.۹۵
۵	۲۴۶۱۳	۲۴۵۶۳	۲۴۶۰۱	۲۴۵۸۷	۲۴۶۰۲.۴۹
۶	۲۵۵۹۱	۲۵۵۰۴	۲۵۲۹۳	۲۵۵۱۲	۲۵۵۴۹.۶۵
۷	۲۳۴۱۰	۲۳۳۶۱	۲۳۲۰۴	۲۳۳۷۱	۲۳۴۱۰
۸	۲۴۲۱۶	۲۴۱۷۳	۲۳۷۶۲	۲۴۱۷۲	۲۴۲۰۲.۴۵
۹	۲۴۴۱۱	۲۴۳۲۶	۲۴۲۵۵	۲۴۳۵۶	۲۴۴۰۸.۴۵
۱۰	۴۲۷۵۷		۴۲۷۰۵	۴۲۷۰۴	۴۲۷۰۵
۱۱	۴۲۵۴۵		۴۲۴۴۵	۴۲۴۵۶	۴۲۴۵۰.۹۵
۱۲	۴۱۹۶۸		۴۱۵۸۱	۴۱۹۳۴	۴۱۹۳۶.۸۵
۱۳	۴۵۰۹۰		۴۴۹۱۱	۴۵۰۵۶	۴۵۰۲۵.۴
۱۴	۴۲۲۱۸		۴۲۰۲۵	۴۲۱۹۴	۴۲۲۰۰.۴۵
۱۵	۴۲۹۲۷		۴۲۶۷۱	۴۲۹۱۱	۴۲۸۹۸.۱۵
۱۶	۴۲۰۰۹		۴۱۷۷۶	۴۱۹۷۷	۴۱۹۸۸.۷
۱۷	۴۵۰۲۰		۴۴۶۷۱	۴۴۹۷۱	۴۵۰۰۷.۹۳
۱۸	۴۳۴۴۱		۴۳۱۲۲	۴۳۳۵۶	۴۳۲۸۶.۲۷
۱۹	۴۴۵۵۴		۴۴۴۷۱	۴۴۵۰۶	۴۴۵۱۵.۷

# سیزدهمین کنفرانس ملی انجمن کامپیوتر ایران

## جزیره کیش، خلیج فارس، ایران ۱۹ الی ۲۱ اسفند ۱۳۸۶

پیشنهادی از سرعت همگرایی به مراتب بالاتری برخوردار است.

### ۵- نتیجه گیری

در این مقاله یک الگوریتم تکرارشونده مبتنی بر اتوماتاهای یادگیر برای حل مساله کوله‌پشتی چندبعدی پیشنهاد شد. نتایج بدست آمده از آزمایشها نشان داد که الگوریتم پیشنهادی برای حل مساله کوله‌پشتی از کارایی بالایی برخوردار است. نتایج شبیه‌سازی‌های انجام گرفته کارایی الگوریتم پیشنهادی را هم از لحاظ کیفیت جوابهای تولید شده و هم از لحاظ سرعت همگرایی به جواب در مقایسه با تعدادی از الگوریتمهای گزارش شده نشان می‌دهد.

جدول ۲ نتایج حاصل برای ۲۰ نمونه تست با ۱۰۰ کالا و ۱۰ کوله‌پشتی را نشان می‌دهد. همانطور که از جدول مشخص است الگوریتم پیشنهادی در اکثر موارد به جوابهای بهتری دست یافته است.

جدول ۳ نتایج حاصل برای ۵ نمونه تست با ۵۰۰ کالا و ۵ کوله‌پشتی را نشان می‌دهد. بهترین نتایج بدست آمده برای این مجموعه در [۲۰] گزارش شده است. در این مقاله برای حل مساله کوله‌پشتی یک الگوریتم ترکیبی که از جستجوی tabu و برنامه‌نویسی خطی استفاده می‌کند، گزارش شده است. نتایج حاصل از الگوریتم MKPLA شامل میانگین ۱۰ بار اجرای الگوریتم با ۱۰۰۰ تکرار در هر اجرا می‌باشد. همان‌طور که از جدول مشخص است الگوریتم پیشنهادی در همه موارد جوابهای بهتری نسبت به الگوریتم Alaya تولید کرده است. به علاوه الگوریتم

جدول (۲): مقایسه نتایج الگوریتم MKPLA با دیگر الگوریتمهای گزارش شده برای ۱۰۰ کالا و ۱۰ کوله‌پشتی

شماره مساله	بهترین جواب شناخته شده	L. & M. (میانگین)	ALAYA C* میانگین	MKPLA C* میانگین
۰	۲۳۰۶۴	۲۲۹۹۶	۲۳۰۱۶	۲۳۰۵۳.۴
۱	۲۲۸۰۱	۲۲۶۷۲	۲۲۷۱۴	۲۲۷۳۸.۱۳
۲	۲۲۱۳۱	۲۱۹۸۰	۲۲۰۳۴	۲۲۰۸۶.۳۸
۳	۲۲۷۷۲	۲۲۶۳۱	۲۲۶۳۴	۲۲۶۱۹.۷۵
۴	۲۲۷۵۱	۲۲۵۷۸	۲۲۵۴۷	۲۲۵۹۶
۵	۲۲۷۷۷	۲۲۵۶۵	۲۲۶۰۲	۲۲۶۲۸.۲۳
۶	۲۱۸۷۵	۲۱۷۵۸	۲۱۷۷۷	۲۱۷۸۸.۸۷
۷	۲۲۶۳۵	۲۲۵۱۹	۲۲۴۵۳	۲۲۵۹۵.۸
۸	۲۲۵۱۱	۲۲۲۹۲	۲۲۳۵۱	۲۲۴۰۸.۴
۹	۲۲۷۰۲	۲۲۵۸۸	۲۲۵۹۱	۲۲۷۰۲
۱۰	۴۱۳۹۵		۴۱۳۲۹	۴۱۲۶۷.۶۵
۱۱	۴۲۳۴۴		۴۲۲۱۴	۴۲۱۶۴.۴۵
۱۲	۴۲۴۰۱		۴۲۳۰۰	۴۲۲۸۴.۲
۱۳	۴۵۶۲۴		۴۵۶۲۱	۴۵۴۱۰.۴
۱۴	۴۱۸۸۴		۴۱۷۳۹	۴۱۷۵۷.۶
۱۵	۴۲۹۹۵		۴۲۹۰۹	۴۲۹۳۸.۲
۱۶	۴۳۵۵۹		۴۳۴۶۴	۴۳۵۳۲.۷۳
۱۷	۴۲۹۷۰		۴۲۹۰۳	۴۲۸۷۲.۴
۱۸	۴۲۲۱۲		۴۲۱۴۶	۴۲۱۴۷.۹۵
۱۹	۴۱۲۰۷		۴۱۰۶۷	۴۱۱۰۳.۴۷

جدول (۳): مقایسه نتایج الگوریتم MKPLA با دیگر الگوریتمهای گزارش شده برای ۵۰۰ کالا و ۵ کوله‌پشتی

شماره مساله	بهترین جواب شناخته شده	ALAYA		MKPLA	
		C*	میانگین	C*	میانگین
۰	۱۲۰۱۳۴	۱۱۹۶۵۸	۸۸۵	۲۰۲	۱۱۹۹۹۲
۱	۱۱۷۸۶۴	۱۱۷۴۲۳	۸۵۷	۳	۱۱۷۶۹۰
۲	۱۲۱۱۱۲	۱۲۰۶۲۲	۸۶۰	۲۱۲	۱۲۰۹۷۶
۳	۱۲۰۸۰۴	۱۲۰۲۷۹	۸۱۴	۱۲۲	۱۲۰۶۵۰.۷
۴	۱۲۲۳۱۹	۱۲۱۸۲۹	۸۲۶	۲	۱۲۲۱۳۳

۷- مراجع

<sup>1</sup> Linear Reward Penalty

<sup>2</sup> Linear Reward Epsilon Penalty

<sup>3</sup> Linear Reward Inaction

- [1] Balas, E., and Zemel, E., *An algorithm for large zero-one knapsack problems*, Operation Research. vol. 28, 1130–1154, 1980.
- [2] Plateau, G., and Elkihel, M., *A hybrid algorithm for the 0-1 knapsack problem*, Methods of Operation Research. vol. 49, pp. 277–293, 1985.
- [3] Pisinger, D., *An exact algorithm for large multiple knapsack problem*. European Journal of Operation Research, vol. 114, pp. 528–541, 1999.
- [4] Pisinger, D., *Algorithms for knapsack problems*. Ph.D. Thesis, DIKU, University of Copenhagen, Report 95/1 1995.
- [5] Lee, J., Shragowitz, E., and Sahni, S., *A hypercube algorithm for the 0/1 knapsack problem*. Journal of Parallel and Distributed Computing, vol. 5, pp. 438–456, 1988.
- [6] Ibarra, O. H., and Kim, C. E., *Fast approximation algorithms for the knapsack problem*. Journal of ACM, vol. 22, pp. 463–468, 1975.
- [7] Riadl, G. R., *Weight-Codings in a genetic algorithm for the multiconstraint knapsack problem*. Proceedings of the 2002 IEEE Congress on Evolutionary Computation, pp. 1564–1569, 2002.
- [8] Riadl, G. R., *An improved genetic algorithm for the multiconstrained 0–1 knapsack problem*. Proceedings of the 5th IEEE Conference on Evolutionary Computation, Anchorage, Alaska, pp. 207–211, May 1998.
- [9] Khuri, S., Back, T., and Heitkotter, J., *The Zero/One multiple knapsack problem and genetic algorithms*. Proceedings of the 1994 ACM symposium of Applied Computation, pp. 188–193, 1994.
- [10] Fidanova, S., *Evolutionary algorithm for multidimensional knapsack problem*. Proceedings of PSNVII, 2002.
- [11] Alaya, I., Solnon, Ch., and Ghedira, K., *Ant algorithm for the multidimensional knapsack problem*. Dans Proceedings of International Conference on Bioinspired Methods and their Applications, Slovenia, 2004.
- [12] Sutton, R. S., and Barto, A. G., *Reinforcement learning: An introduction*. MA: MIT Press, Cambridge, 1998.
- [13] Thathachar, M. A. L., Sastry, P. S., *Varieties of learning automata: An overview*. IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics, 32, 6, 2002.
- [14] Pirkul, H., *A Heuristic solution procedure for the multiconstrained zero-one knapsack problem*. Naval Research Logistics 34, pp. 161–172, 1987.
- [15] Chu, P. C., *A genetic algorithm approach for combinatorial optimization problems*. Ph.D. Thesis at the Management School, Imperial College of Science, London, 1997.
- [16] Chu, P. C., and Beasley, J. E., *A genetic algorithm for the multidimensional knapsack problem*. Journal of Heuristics 4, pp. 63–86, 1998.
- [17] Beasley, J. E., *OR-Library: Distributing test problems by electronic mail*. Journal of Operational Research society 41, 1069–1072, 1990.
- [18] Leguizamón, G., and Michalewicz, Z., *A new version of Ant System for subset problem*. Proceedings of Congress on Evolutionary Computation, pp. 1459–1464, 1999.
- [19] Vasquez, M., and Hao, J. K., *A hybrid approach for the 0-1 multidimensional knapsack problem*. IJCAI-01, Washington, 2001.