Un algoritm de optimizare a alocării rezultatelor tranzacțiilor bursiere(*)

Claudiu Vințe
Goldman Sachs Japan Holdings, Ltd.

Abstract. *In this paper, I wish to propose the Integer Allocation employing Tabu Search in conjunction with Simulated Annealing Heuristics for optimizing the distribution of trading executions in investors' accounts.*

There is no polynomial algorithm discovered for Integer Linear Programming *(a problem which is NP-complete). Generally, the practical experience shows that large-scale integer linear programs seem as yet practically unsolvable or extremely time-consuming. The algorithm described herein proposes an alternative approach to the problem.*

The algorithm consists of three steps: allocate the total executed quantity proportionally on the accounts, based on the allocation instructions *(*pro-rata *basis); construct an initial solution, distributing the executed prices; improve the solution iteratively, employing* Tabu Search *in conjunction with* Simulated Annealing *heuristics.*

Key words: integer allocation; allocation instructions; pro-rata coefficient; optimization.

Problema practică de alocare în numere întregi pentru care s-a conceput algoritmul ce va fi prezentat în acest articol provine din activitatea bursieră realizată în mod curent de o casă de brokeraj. Pentru anumite instrumente financiare (contracte *futures* și *options,* de exemplu) și/sau în anumite legislații bursiere (cazul sistemului bursier Japonez) multiplele prețuri la care se poate executa (întâlni cererea cu oferta la bursă) un ordin de vânzare/cumpărare a unui instrument financiar nu pot fi consolidate într-un preț mediu ponderat unic pentru întreg ordinul, ci trebuie ca aceste execuții individuale să fie împărțite și distribuite în conturile clientului (investitorului). Această împărțire și distribuire trebuie să se facă de așa natură încât să se respecte dimensiunea lotului specifică unui instrument financiar dat: numărul minim de acțiuni, contracte etc. care se poate trimite către automatul bursier și care este prestabilit pentru fiecare instrument financiar care se tranzacționează la o anumită bursă.

Concret, problema are următoarea formulare: dându-se un set de *n* conturi pe care un investitor pe piața bursieră le are deschise la o casă de brokeraj și pe care dorește să le utilizeze pentru tranzacția considerată, tabloul de mai jos descrie maniera în care investitorul dorește să distribuie în aceste *n* conturi cantitatea totală a ordinului său de vânzare sau de cumpărare $Q = \sum_{i=1}^{n} q_i$ a unui anumit instrument financiar

$$\begin{bmatrix} A_1 & A_2 & A_3 & \cdots & A_n \\ q_1 & q_2 & q_3 & \cdots & q_n \end{bmatrix}.$$

(*) Goldman Sachs Group în New York a depus la data de 3 aprilie 2006 o cerere de patent pentru revendicarea priorității inventării algoritmului descris în acest articol.

Această matrice poartă, de asemenea, denumirea de *instrucțiuni de alocare.*

Algoritmul care va fi prezentat în continuare propune o abordare euristică în rezolvarea unei probleme specifice de programare liniară în numere întregi, și anume alocarea cât mai echitabilă – din perspectiva prețului mediu pe fiecare cont – pe conturile înregistrate de un investitor la o casă de brokeraj a ordinelor executate la bursă.

La bursă, totalul cantității ordonate de client (Q) poate să fie executat în mai multe porțiuni (tranșe), la prețuri diferite:

$$\begin{bmatrix} (p_1, e_1) \\ (p_2, e_2) \\ (p_3, e_3) \\ \vdots \\ (p_m, e_m) \end{bmatrix},$$

unde:

(p_i, e_i), $i = \overline{1, m}$ este perechea care desemnează cantitatea de ordine executate în cazul unei tranșe (e_i) la prețul (p_i), *m* fiind numărul de execuții (porțiuni, tranșe) la prețuri diferite primite de la bursă, pentru ordinul plasat de casa de brokeraj în numele investitorului. De remarcat faptul că este satisfăcută următoarea condiție:

$$0 \le \sum_{i=1}^{m} e_i = E \le Q = \sum_{j=1}^{n} q_j ,$$

unde

E este numărul total de ordine executate la bursă.

În consecință, ordinul poate fi *executat integral, executat parțial* sau poate să rămână *neexecutat*.

În acest context, problema de alocare în numere întregi constă în a găsi o astfel de distribuție *D*, constând din multipli întregi de loturi din ordinele executate la bursă, în conturile investitorului, astfel încât, respectându-se *instrucțiunile de alocare*, să se obțină un preț mediu ponderat pe fiecare cont cât mai apropiat de prețul mediu ponderat general – obținut pe baza întregii cantități executate la bursă la diferite prețuri, din ordinul considerat. Formalizând, este vorba de minimizarea următoarei funcții obiectiv: $O = \sum_{j=1}^{n} \left| \left(\bar{p} - \overline{p_j} \right) \right|$.

unde:

$\left| \left(\bar{p} - \overline{p_j} \right) \right|$ este valoare absolută a diferenței $\left(\bar{p} - \overline{p_j} \right)$,

$\overline{p_j} = \dfrac{\sum_{i=1}^{m} (p_i \times a_{ij})}{\sum_{i=1}^{m} a_{ij}}$ este prețul mediu ponderat la nivel de cont

și $\bar{p} = \dfrac{\sum_{i=1}^{m} (p_i \times e_i)}{\sum_{i=1}^{m} e_i}$ este prețul mediu ponderat general.

$$\begin{bmatrix} & A_1 & A_2 & \cdots & A_n \\ (p_1, e_1) & a_{11} & a_{12} & \cdots & a_{1n} \\ (p_2, e_2) & a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (p_m, e_m) & a_{m1} & a_{m2} & \cdots & a_{mn} \\ \bar{p} & \overline{p_1} & \overline{p_2} & \cdots & \overline{p_n} \end{bmatrix}.$$

Prezentare

Algoritmul propus constă în trei faze:

1. determinarea numărului de ordine executate ce trebuie alocate, în mod proporțional pe fiecare cont, din numărul total de ordine executate, pe baza *instrucțiunilor de alocare*;

2. construirea unei soluții inițiale, prin distribuirea efectivă pe conturi a ordinelor executate, în funcție de prețurile de execuție;

3. îmbunătățirea soluției problemei de alocare în mod iterativ, prin utilizarea de euristici de *căutare tabu* și *călire simulată.*

Faza 1 – distribuirea ordinelor totale executate pe conturi, în mod proporțional

Această fază începe prin determinarea pentru fiecare cont a coeficientului de repartizare proporțională. Acești coeficienți reprezintă proporția din cantitatea totală a ordinului care trebuie să revină fiecărui cont, pe baza *instrucțiunilor de alocare* furnizate chiar de client (investitor) casei de brokeraj, la momentul plasării ordinului:

$$0 \le c_j = \frac{q_j}{\sum_{j=1}^{n} q_j} \le 1, \; j = \overline{1, n}.$$

Determinăm apoi, pentru fiecare cont, procentul de umplere – procentul până la care cantitatea cerută în fiecare cont este satisfăcută prin distribuirea proporțională a ordinelor executate:

$$f_j = \frac{\left(q_j - \langle c_j \rangle\right)}{q_j} \times 100,\ j = \overline{1, n},$$

unde:

$\langle c_j \rangle$ este partea întreagă a lui c_j (trunchiere).

În aceste condiții, cantitatea rămasă (reziduală) a fi satisfăcută pentru un cont se calculează astfel:

$$r_j = q_j - \langle c_j \rangle,\ j = \overline{1, n}.$$

Totalul cantității reziduale este dat de:

$$R = \sum_{j=1}^{n} \left(q_j - \langle c_j \rangle\right) = \sum_{j=1}^{n} r_j.$$

De asemenea, fiecare cont poate avea asignată o anumită prioritate $t_j, j = \overline{1, n},$ care intră în funcțiune în cazul în care două sau mai multe conturi au de satisfăcut cereri de ordine executate egale.

Tabloul următor unifică toate corespondențele prezentate mai sus.

$$\begin{bmatrix} A_1 & A_2 & A_3 & \cdots & A_n \\ q_1 & q_2 & q_3 & \cdots & q_n \\ & & & & \\ c_1 & c_2 & c_3 & \cdots & c_n \\ f_1 & f_2 & f_3 & \cdots & f_n \\ r_1 & r_2 & r_3 & \cdots & r_n \\ & & & & \\ t_1 & t_2 & t_3 & \cdots & t_n \end{bmatrix}$$

Descriem în continuare algoritmul de repartizare proporțională pe conturi a numărului total de ordine executate.

determină $c_j,\quad j = \overline{1, n}$

distribuie cantitatea totală executată *E* în mod proporțional pe conturi

$$A_j \leftarrow E \times c_j \quad j = \overline{1, n}$$

determină $f_j, r_j \quad j = \overline{1, n}$

$$R \leftarrow \sum_{i=1}^{n} r_j$$

while $R > 0$

determină contul care va primi unul din loturile rămase ca și reziduu după repartizarea cu trunchiere, folosind următorul criteriu compus $(f_j, r_j, t_j) \quad j = \overline{1, n}$

$$k \leftarrow \min_{j=1,\,n}\left\{ t_j \otimes \max_{j=1,\,n}\left\{ r_j \otimes \min_{j=1,\,n}\left\{ f_j \right\} \right\} \right\}$$

$A_k \leftarrow A_k + ls$ (unde *ls* este dimensiune unui lot, specifică instrumentului financiar considerat)

determină f_k, r_k

$R \leftarrow R - ls$

Faza 2 – construirea unei soluții inițiale

Ajunși la acest punct al algoritmului avem determinată cantitatea ce va fi alocată în fiecare cont, după distribuirea întregii cantități reziduale. Pentru a construi o soluție de start trebuie să aranjăm datele într-o manieră care să fie consistentă cu funcția obiectiv propusă. În acest sens, vom ordona prețurile execuțiilor venite de la bursă în ordine crescătoare, funcție de distanța, în valoare absolută, la care se găsesc față de prețul mediu ponderat general $\bar{p}$, $l_i = \left| \left(p_i - \bar{p}\right) \right|,\ i = \overline{1, m}.$

Vom construi apoi două subserii de prețuri, în felul următor: cele care sunt mai mici sau egale cu prețul mediu general $\bar{p}$ și cele care sunt mai mari decât media generală.

$$U = \left(p_k : p_k \le \bar{p},\quad 1 \le k \le m\right)$$
$$V = \left(p_{k'} : p_{k'} > \bar{p},\quad 1 \le k' \le m\right)$$

Strategia urmărită este de a aloca cel mai apropiat preț de media generală conturilor care au cea mai mică cerere, în timp ce conturile care necesită o cantitate mai mare vor avea o probabilitate mai ridicată de a primi loturi din cantitatea executată la diferite prețuri, astfel încât acestea să se compenseze reciproc. Dacă sunt mai multe conturi cu aceeași cerere, atunci fiecare din aceste conturi va primi câte un lot din cantitatea executată la prețul considerat, până când această cantitate este epuizată.

Algoritmul este următorul:

determină $l_i = \left| \left(p_i - \bar{p}\right) \right|,\ i = \overline{1, m}.$

for $j \leftarrow 1$ *to* n

inițializează prețul mediu pe cont cu cel mai mic preț la care încă este o cantitate disponibilă

$$\overline{p_j^c} \leftarrow \min_{i=1,\,m}\{l_i;\, e_i > 0\}$$

determină prețul care afectează cel mai puțin prețul mediu al contului, căutând în ambele subserii U și V

$$\overline{p_j^n} \leftarrow \min_{\substack{p_k \in U \\ p_k \in V}} \left\{ \frac{\left(\overline{p_j^c} \times a_j^c\right) + (p_k \times ls)}{a_j^c + ls} \right\}$$

(unde *ls* este dimensiunea lotului instrumentului financiar tranzacționat,

$a_j = \sum_{i=1}^{m} a_{ij}$ este cantitatea totală alocată deja în contul A_j, iar $\overline{p_j^n}$, $\overline{p_j^c}$ reprezintă prețul nou și, respectiv, cel curent, determinate pentru contul considerat)

$$a_{kj}^n \leftarrow a_{kj}^c + ls$$

(unde a_{kj}^n este noua cantitate alocată în contul A_j la prețul p_k),

$$j \in \left\{ x : q_j = q_x,\ 1 \le x \le n \right\}$$

$a_j^n \leftarrow a_j^c + ls$ (unde a_j^n este noua cantitate *totală* alocată în contul A_j),

$$j \in \left\{ x : q_j = q_x,\ 1 \le x \le n \right\}$$

$$e_k \leftarrow e_k - ls$$

(unde e_k este cantitatea rămasă disponibilă la prețul p_k).

Faza 3 – Optimizarea, îmbunătățirea soluției inițiale

La acest stadiu al algoritmului deținem deja o soluție pentru problema de alocare în numere întregi, și această soluție o vom folosi ca și soluție inițială pentru faza de optimizare.

Obiectivul acestei faze este de a îmbunătăți soluția problemei de alocare prin determinarea acelor permutări de loturi între acele conturi și la acele prețuri astfel încât să realizăm minimizarea funcției obiectiv, definită mai jos, fără a altera cantitatea totală alocată pe fiecare cont:

$$O = \min\left\{ \sum_{j=1}^{n} \left| \Delta_j \right| \right\},$$

unde:

$$\Delta_j = \frac{\overline{p}_j}{\overline{p}} - 1 \quad (j = \overline{1, n}),\ \text{și}$$

$\left|\Delta_j\right|$ este valoarea absolută pentru Δ_j (Cook & alții, 1998).

$$\begin{bmatrix}
 & A_1 & A_2 & \cdots & & & & \cdots & A_n \\
 & & & & & & & & \\
(p_1, e_1) & a_{11} & a_{12} & \cdots & & & & \cdots & a_{1n} \\
(p_2, e_2) & a_{21} & a_{22} & \cdots & & & & \cdots & a_{2n} \\
\vdots & \vdots & \vdots & & \vdots & & \vdots & & \vdots \\
 & & & \cdots & \text{fromX} & \rightarrow & \text{toY} & \cdots & \\
D = & & & & \vdots & & \vdots & & \\
 & & & \cdots & \text{toX} & \leftarrow & \text{fromY} & \cdots & \\
\vdots & \vdots & \vdots & & \vdots & & \vdots & & \vdots \\
(p_m, e_m) & a_{m1} & a_{m2} & \cdots & & & & \cdots & a_{mn} \\
 & & & & & & & & \\
\overline{p} & \overline{p_1} & \overline{p_2} & \cdots & & & & \cdots & \overline{p_n} \\
 & \Delta_1 & \Delta_2 & \cdots & & & & \cdots & \Delta_n
\end{bmatrix}$$

Algoritmul caută să identifice o pereche de conturi între care poate fi interschimbată o cantitate de un lot din instrument financiar considerat. Acest interschimb se urmărește a fi făcut de așa natură încât să conducă căutarea către soluția optimă sau către o soluție în vecinătatea optimului (Kreher, 1999).

$$O_{best} \leftarrow \max\{\text{double}\}$$

determină valoarea delta curentă pentru fiecare cont

$$\Delta_j^c = \frac{\overline{p}_j}{\overline{p}} - 1 \quad (j = \overline{1, n})$$

$$O_c \leftarrow \sum_{j=1}^{n} \left| \Delta_j^c \right|$$ **(valoarea curentă a funcției obiectiv)**

determină

$$\{(\min \text{Row}, \min \text{Col}), (\max \text{Row}, \max \text{Col})\}$$

if *interschimb este posibil*
  while *nu-i optim*
    execută interschimb și determină noile valori delta $\Delta_j^n \quad (j=\overline{1,n})$

    $O_n \leftarrow \sum_{j=1}^{n} \left| \Delta_j^n \right|$ **(noua valoare a funcției obiectiv)**

    if $O_n \geq O_c$
      if $O_c \leq O_{best}$
      $O_{best} \leftarrow O_c$, **renunță la istoria valorilor funcției obiectiv**
      if $(O_n - O_{best}) <$ *Temperatura de Călire*
        if ***Temperatura de Călire este ajustată problemei concrete***
          ridică ***Temperatura de Călire***
        else
          ajustează ***Temperatura de Călire*** **și execută un proces de răcire**
      else
        optim ← true
    else if $O_n \leq O_{best}$
      $O_{best} \leftarrow O_n$, **șterge istoria valorilor funcției obiectiv**
    if *optim*
      mută înapoi cantitățile interschimbate anterior
    else
      $O_c \leftarrow O_n$
      adaugă mutările executate la lista ***Tabu***
      șterge din memoria Tabu mutările mai vechi decât adâncimea de memorie aleasă
      memorează informațiile legate de valoarea curentă a funcției obiectiv
      determină $\{(\min \text{Row}, \min \text{Col}), (\max \text{Row}, \max \text{Col})\}$
      if *interschimb este posibil*
        ridică *Temperatura de Călire*
      else
        optim ← true

if $O_{best} < O_c$
se merge înapoi în istoria valorilor funcției obiectiv pentru regăsirea celei mai bune soluții atinse.

Trebuie remarcat faptul că valoarea funcției obiectiv nu trebuie în mod necesar să fie mai mică la fiecare iterație, față de iterația precedentă (cum se întâmplă în cazul unei simple strategii *Greedy*). Abordarea prin prisma euristicii de *Călire Simulată* permite algoritmului să relaxeze procesul de căutare în spațiul soluțiilor și să evite în acest fel capcanele întinse de optimurile locale. Prin relaxarea condiției de căutare (călire), mersul prin spațiul soluțiilor este redirecționat în așa fel încât să se ofere căutării posibilitatea de a ajunge în proximitatea optimului global (Michalewicz și Fozel, 2000).

Cu toate acestea, introducerea mecanismului de relaxare a condiției de căutare deschide posibilitatea apariției ciclării. Însă, prin introducerea unei *Liste Tabu,* procesul de căutare este astfel dotat cu o memorie de scurtă durată (a se vedea considerațiile teoretice despre adâncimea acestei memorii), aceasta prevenind întoarcerile pe ramurile arborelui de căutare care au fost deja explorate (Glover și Laguna, 2002). Această *Listă Tabu* păstrează, pentru fiecare interschimb realizat, o triadă de felul următor: *(contul DE UNDE, contul CĂTRE CARE, prețul LA CARE).*

Unul dintre aspectele esențiale ale algoritmului este determinarea perechii de conturi între care se poate face un interschimb la un anumit pas și prețurile la care se realizează interschimbul unui lot din cantitatea de produs financiar avut în vedere. Se utilizează în acest sens o strategie *Min-Max*. În mod practic, la fiecare iterație, încercăm să interschimbăm un lot de cantitate între contul care are cel mai mic preț și contul care are cel mai mare preț mediu ponderat, determinate până la acel pas (aceasta este componenta *Max* a strategiei). Partea de *Min* a strategiei intervine în momentul când trebuie să selectăm prețurile la care realizăm interschimbul (de exemplu, încercăm să identificăm acea pereche de prețuri la care interschimbul afectează în cea mai mică măsură prețul mediu al celor două conturi considerate), se realizează în acest fel ajustări fine ale prețurilor medii ponderate ale conturilor investitorului.

Dacă $\left(\max_{j=1,n} \{\Delta_j\} - \min_{j=1,n} \{\Delta_j\} \right) < \varepsilon \ (0 < \varepsilon < 1)$ sau dacă nu se mai poate găsi o pereche de conturi sau o pereche de prețuri care să susțină un interschimb, atunci algoritmul se oprește și cea mai bună soluție detectată până la acel pas devine soluția pe care algoritmul o furnizează drept rezultat.

Așa cum este cazul cu majoritatea algoritmilor euristici, determinarea soluției optime nu este în mod necesar garantată. Cu toate acestea, testele și rezultate implementării practice ale algoritmului au arătat cu consistență îmbunătățiri semnificative ale calității soluției obținute după cea de-a treia fază, comparativ cu soluția inițială construită în faza a doua a alocării.

Bibliografie

Cook, W.J., Cunningham, W.H., Pulleyblank, W.R., Schrijver, A. (1998). *Combinatorial Optimization* – John Wiley & Sons, Inc.

Glover, F., Laguna, M. (2002). *Tabu Search* – Kluwer Academic Publishers, Sixth Printing

Kreher, D.L., Douglas, R.S, (1999). *Combinatorial Algorithms – Generation, Enumeration and Search* – CRC Press LLC

Michalewicz, Z., David, B.F. (2000). *How to Solve It: Modern Heuristics* – Springer – Verlag Berlin Heidelberg.

Nemhauser, G., Wolsey, L. (1999). *Integer and Combinatorial Optimization* – John Wiley & Sons, Inc.

Schrijver, A. (1986). *Theory of Linear and Integer Programming* – John Wiley & Sons Ltd

Vințe, C. (2001). *The Proceedings of the Fifth International Symposium on Economic Informatics, Upon An Integer Allocation Problem* – Editura Economică, Inforec Printing House