A Graphical Algorithm for Solving an Investment Optimization Problem

Evgeny R. Gafarov, Alexandre Dolgui, Alexander Lazarev, Frank Werner

MISTA 2013 manuscript No.
(will be inserted by the editor)

A Graphical Algorithm for Solving an Investment Optimization Problem

Evgeny R. Gafarov · Alexandre Dolgui · Alexander A. Lazarev · Frank Werner.

the date of receipt and acceptance should be inserted later

Abstract In this paper, a graphical algorithm (GrA) is presented for an investment optimization problem. This algorithm is based on the same Bellman equations as the best known dynamic programming algorithm (DPA) for the problem but the GrA has several advantages in comparison with the DPA. Based on this GrA, a fully-polynomial time approximation scheme is proposed having the best known running time. The idea of the GrA presented can also be used to solve some similar scheduling or lot-sizing problems in a more effective way.

1 Introduction

The Project Investment Problem can be formulated as follows. A set N of n potential projects and an investment budget (amount) $A > 0,\ A \in Z$, are given. For each project $j,\ j = 1, \ldots, n$, a profit function $f_j(x),\ x \in [0, A]$, is given, where the value $f_j(x')$ denotes the profit received if the amount x' is invested into the project j. The objective is to determine an amount $x_j \in [0, A],\ x_j \in Z$, for each project $j \in N$ such that $\sum_{j:=1}^{n} x_j \leq A$ and the total profit $\sum_{j:=1}^{n} f_j(x_j)$ is maximized.

E.R. Gafarov
Institute of Control Sciences of the Russian Academy of Sciences, Profsoyuznaya st. 65, 117997 Moscow, Russia
E-mail: axel73@mail.ru

A. Dolgui
Ecole Nationale Superieure des Mines, CNRS UMR6158, LIMOS, F-42023 Saint-Etienne, France
E-mail: dolgui@emse.fr

A.A. Lazarev
Lomonosov Moscow State University, Higher School of Economics (National Research University), Institute of Control Sciences of the Russian Academy of Sciences, Profsoyuznaya st. 65, 117997 Moscow, Russia,
E-mail: jobmath@mail.ru

F. Werner
Fakultät für Mathematik, Otto-von-Guericke-Universität Magdeburg, PSF 4120, 39016 Magdeburg, Germany,
E-mail: frank.werner@ovgu.de

In this paper, we deal with piecewise linear functions $f_j(x)$. Suppose that the interval $[0, A]$ can be written as

$$[0, A] = [t_j^0, t_j^1] \bigcup (t_j^1, t_j^2] \bigcup \ldots \bigcup (t_j^{k-1}, t_j^k] \bigcup \ldots \bigcup (t_j^{k_j-1}, t_j^{k_j}]$$

such that the profit function has the form $f_j(x) = b_j^k + u_j^k(x - t_j^{k-1})$, if $x \in (t_j^{k-1}, t_j^k]$, where k is the number of the interval, b_k^j is the value of the function at the beginning of the interval, and u_j^k is the slope of the function. Without loss of generality, assume that $b_j^1 \leq b_j^2 \leq \ldots \leq b_j^{k_j}$ and $t_j^k \in Z,\ j \in N,\ k = 1, 2, \ldots, k_j$, and that $t_j^{k_j} = A,\ j = 1, 2, \ldots, n$.

A special case of this problem is similar to the well-known bounded knapsack problem:

$$\begin{array}{ll} \text{maximize} & \sum_{j:=1}^{n} p_j x_j \\ s.t. & \sum_{j:=1}^{n} w_j x_j \leq A, \\ & x_j \in [0, b_j],\ x_j \in Z,\ j = 1, 2, \ldots, n, \end{array} \tag{1}$$

for which a dynamic programming algorithm (DPA) of time complexity $O(nA)$ is known [3].

The following problem is also similar to the problem under consideration:

$$\begin{array}{ll} \text{minimize} & \sum_{j:=1}^{n} f_j(x_j) \\ s.t. & \sum_{j:=1}^{n} x_j \geq A, \\ & x_j \in [0, A],\ x_j \in Z,\ j = 1, 2, \ldots, n, \end{array} \tag{2}$$

where $f_j(x_j)$ are piecewise linear as well. For this problem, a DPA with a running time of $O(\sum k_j A)$ [4] and a fully polynomial-time approximation scheme (FPTAS) with a running time of $O((\sum k_j)^3/\varepsilon)$ [5] are known.

In this paper, we present an alternative solution algorithm with a running time of $O(\sum k_j A)$ and an FPTAS based on this solution algorithm with a running time of $O(\sum k_j n \log\log n/\varepsilon)$.

The remainder of the paper is as follows. In Section 2, we present the Bellman equations to solve the problem under consideration. In Section 3, a graphical algorithm (GrA) based on an idea from [1] is presented. In Section 4, an FPTAS based on this GrA is derived.

2 Dynamic programming algorithm

In this section, we present a DPA for the problem considered. For any project j and any state $t \in [0, A]$, we define $F_j(t)$ as the maximal profit incurred for the projects $1, 2, \ldots, j$, when the remaining budget available for the projects $j + 1, j + 2, \ldots, n$ is equal to t. Thus, we have:

$$\begin{array}{ll} F_j(t) = \max & \sum_{h:=1}^{j} f_h(x_h) \\ s.t. & \sum_{h:=1}^{j} x_h \leq A - t, \\ & x_h \geq 0,\ x_h \in Z,\ h = 1, 2, \ldots, j. \end{array} \tag{3}$$

We define $F_j(t) = 0$ for $t \notin [0, A]$. Then we have the following recursive equations:

$$F_j(t) = \max_{x\in[0,A-t]}\{f_j(x) + F_{j-1}(t+x)\}$$
$$= \max_{1\le k\le k_j} \max_{x\in(t_j^{k-1},t_j^k]\bigcap[0,A-t]} \{b_j^k - u_j^k t_j^{k-1} + u_j^k \cdot x + F_{j-1}(t+x)\}, \quad (4)$$
$$j = 1, 2, \ldots, n.$$

Lemma 1 *All functions $F_j(t),\ j = 1, 2, \ldots, n$, are non-increasing on the interval $[0, A]$.*

The proof of this lemma immediately follows from the definition of the functions $F_j(t)$.

The running time of the DPA using such a type of Bellman equations is $O(\sum k_j A)$ if we use an idea from [4].

3 Graphical algorithm

In this section, we present a GrA which constructs the functions $F_j(t),\ j = 1, 2, \ldots, n$, in a more effective way in comparison with the DPA. Below we prove that the functions $F_j(t),\ j = 1, 2, \ldots, n$, constructed in the GrA are piecewise linear (see Lemma 4).

Any piecewise linear function $\varphi(x)$ considered in this paper can be defined by three sets of numbers: a set of break points I (at each break point, a new linear segment of the piecewise linear function begins), a set of slopes U and a set of values of the function at the beginning of the interval B. Let the notation $I[k]$ denote the k-th element in the ordered set I. The same notations will be used for the sets U and B as well. The notation $\varphi.I[k]$ denotes the k-th element of the set I of the function $\varphi(x)$. Then, for example, for $x \in (t_j^{k-1}, t_j^k] = (f_j.I[k-1], f_j.I[k]]$, we have

$$f_j(x) = f_j.B[k] + f_j.U[k](x - f_j.I[k-1]).$$

Note that

$$\varphi.I[k] < \varphi.I[k+1],\ k = 1, \ldots, |\varphi.I| - 1 \quad \text{and} \quad k_j = |f_j.I|.$$

In each step $j,\ j = 1, 2, \ldots, n$, of the subsequent algorithm, the temporary piecewise linear functions Ψ_j^i and Φ_j^i are constructed. Recall that the functions $F_j(t),\ j = 1, 2, \ldots, n$, constructed in the GrA are piecewise linear as well. For $t \in Z$, their values are equal to the values of the functions $F_j(t)$ considered in the DPA.

Let $\varphi.I[-1] = 0$ and $\varphi.I[|\varphi.I| + 1] = A$. Remind that $\varphi.I[|\varphi.I|] = A$.
The points $t \in \varphi.I$ and the other end points of the intervals with the piecewise linear functions considered in this article will be called *break points*. To *construct a function* in the GrA means to compute their sets I, U and B.

Graphical algorithm

1. Let $F_0(t) = 0$, i.e., $F_0.I := \{A\},\ F_0.U := \{0\},\ F_0.B := \{0\}$;
2. FOR $j := 1$ TO n DO
   2.1. FOR $k := 1$ TO k_j DO
      2.1.1. Construct the temporary function

$$\Psi_j^k(t) = f_j.B[k] - f_j.U[k] \cdot f_j.I[k-1] + f_j.U[k] \cdot t + F_{j-1}(t)$$

according to Procedure 2.1.1.;

2.1.2. Construct the temporary function

$$\Phi_j^k(t) = \max_{x \in (f_j.I[k-1], f_j.I[k]] \bigcap [0, A-t]} \{\Psi_j^k(t+x) - f_j.U[k] \cdot t\}$$

according to Procedure 2.1.2.;

2.1.3. IF $k = 1$ THEN $F_j^k(t) := \Phi_j^k(t)$ ELSE $F_j^k(t) := \max\{F_j^{k-1}(t), \Phi_j^k(t)\}$.

2.2. $F_j(t) := F_j^{k_j}(t)$. Modify the sets I, U, B of the function $F_j(t)$ according to Procedure 2.2.

3. The optimal objective function value is equal to $F_n(0)$.

Procedure 2.1.1.

Given are k and j;
$\Psi_j^k.I = \emptyset$, $\Psi_j^k.U = \emptyset$ and $\Psi_j^k.B = \emptyset$.
FOR $i := 1$ TO $|F_{j-1}.I|$ DO
add the value $F_{j-1}.I[i]$ to the set $\Psi_j^k.I$;
add the value

$$f_j.B[k] - f_j.U[k] \cdot f_j.I[k-1] + f_j.U[k] \cdot F_{j-1}.I[i] + F_{j-1}.B[i]$$

to the set $\Psi_j^k.B$;
add the value $f_j.U[k] + F_{j-1}.U[i]$ to the set $\Psi_j^k.U$;

In Procedure 2.1.1., we shift the function $F_{j-1}(t)$ up by the value $f_j.B[k] - f_j.U[k] \cdot f_j.I[k-1]$ and increase all slopes in its diagram by $f_j.U[k]$. If all values $t \in F_{j-1}.I$ are integer, then all values from the set $\Psi_j^i.I$ are integer as well. It is obvious that Procedure 2.1.1. can be performed in $O(|F_{j-1}.I|)$ time.

Before describing Procedure 2.1.2., we present Procedure FindMax in which the maximum function $\varphi(t)$ of two linear fragments $\varphi_1(t)$ and $\varphi_2(t)$ is constructed.

Procedure FindMax

1. Given are the functions $\varphi_1(t) = b_1 + u_1 \cdot t$ and $\varphi_2(t) = b_2 + u_2 \cdot t$ and an interval $(t', t'']$. Let $u_1 \leq u_2$;
2. IF $t'' - t' \leq 1$ THEN RETURN $\varphi(t) = \max\{\varphi_1(t''), \varphi_2(t'')\}$ defined on the interval $(t', t'']$;
3. Find the intersection point t^* of $\varphi_1(t)$ and $\varphi_2(t)$;
4. IF t^* does not exist OR $t^* \notin (t', t'']$ THEN
   IF $b_1 + u_1 \cdot t' > b_2 + u_2 \cdot t'$ THEN RETURN $\varphi(t) = \varphi_1(t)$ defined on the interval $(t', t'']$;
   ELSE RETURN $\varphi(t) = \varphi_2(t)$ defined on the interval $(t', t'']$;
5. ELSE
   IF $t^* \in Z$ THEN
      $\varphi(t) := \varphi_1(t)$ on the interval $(t', t^*]$;
      $\varphi(t) := \varphi_2(t)$ on the interval $(t^*, t'']$;
      RETURN $\varphi(t)$;
   ELSE IF $t^* \notin Z$ THEN
      $\varphi(t) := \varphi_1(t)$ on the interval $(t', \lfloor t^* \rfloor]$;
      $\varphi(t) := b_2 + u_2 \cdot \lfloor t^* \rfloor$ on the interval $(\lfloor t^* \rfloor - 1, \lfloor t^* \rfloor]$;
      $\varphi(t) := \varphi_2(t)$ on the interval $(\lfloor t^* \rfloor, t'']$;

RETURN $\varphi(t)$;

If both points t' and t'' are integer, then $\varphi.I$ contains only integer break points t. The running time of Procedure FindMax is constant.

In Procedure 2.1.2., we do the following. When we shift s' to the right, we shift the interval $T' = [t_{left}, t_{right}]$ of the length $f_j.I[k] - f_j.I[k-1]$. We have to use the values $\Psi_j^k(x)$ for $x \in T'$ to calculate $\Phi_j^k(t)$ at the point $t = s'$. Since $\Psi_j^k(x)$ is piecewise linear, it is only necessary to consider the values $\Psi_j^k(x)$ at the break points belonging to T' and at the end points of the interval T'. So, if we shift s' to the right by a small value $x \in [0, \varepsilon]$ such that all the break points remain the same, then the value $\Phi_j^k(t)$ will be changed according to the value $\varphi_{max}(x)$.

So, in cycle [2.1.2.5], we shift s' to the right. In steps [2.1.2.6]-[2.1.2.8], we look for the first and the last break point belonging to the current interval T'. For these two break points found and the best found break point with an index $s \in \{v, v+1, \ldots, w\}$ (step [2.1.2.14]), we construct the functions $\varphi_{left}(x), \varphi_{right}(x), \varphi_{inner}(x)$ and their maximum function $\varphi_{max}(x)$ (see steps [2.1.2.15]-[2.1.2.18]) according to which the value $\Phi_j^k(t)$ is changed when we shift s' to the right by a small value $x \in [0, \varepsilon]$. In steps [2.1.2.19]-[2.1.2.23], we construct the function $\Phi_j^k(t)$ based on the function $\varphi_{max}(x)$.

Procedure 2.1.2.

2.1.2.1. Given are k, j and $\Psi_j^k(t)$;
2.1.2.2. $\Phi_j^k.I := \emptyset$, $\Phi_j^k.U := \emptyset$ and $\Phi_j^k.B := \emptyset$;
2.1.2.3. $s' := 0$, $t_{left} := s' + f_j.I[k-1]$, $t_{right} := \min\{s' + f_j.I[k], A\}$;
2.1.2.4. Let $T' = \{\Psi_j^k.I[v], \Psi_j^k.I[v+1], \ldots, \Psi_j^k.I[w]\}$ be the maximal subset of $\Psi_j^k.I$, where $t_{left} < \Psi_j^k.I[v] < \ldots < \Psi_j^k.I[w] < t_{right}$,
Let $T := \{t_{left}\} \bigcup T' \bigcup \{t_{right}\}$;
2.1.2.5. WHILE $s' \leq A$ DO
2.1.2.6. IF $T' = \emptyset$ THEN let
$w + 1 = argmax_{i=1,2,\ldots,|\Psi_j^k.I|}\{\Psi_j^k.I[i] | \Psi_j^k.I[i] > t_{right}\}$
and $v = argmax_{i=1,2,\ldots,|\Psi_j^k.I|}\{\Psi_j^k.I[i] | \Psi_j^k.I[i] > t_{left}\}$;
2.1.2.7. IF $w+1$ is not defined THEN let $w + 1 = |\Psi_j^k.I|$;
2.1.2.8. IF v is not defined THEN let $v = |\Psi_j^k.I|$;
2.1.2.9. IF $t_{left} < A$ THEN $\varepsilon_{left} := \Psi_j^k.I[v] - t_{left}$ ELSE $\varepsilon_{left} := A - s'$;
2.1.2.10. IF $t_{right} < A$ THEN $\varepsilon_{right} := \Psi_j^k.I[w+1] - t_{right}$ ELSE $\varepsilon_{right} := +\infty$;
2.1.2.11. $\varepsilon := \min\{\varepsilon_{left}, \varepsilon_{right}\}$;
2.1.2.12. IF $t_{left} < A$ THEN

$$b_{left} := \Psi_j^k.B[v] + \Psi_j^k.U[v] \cdot (t_{left} - \Psi_j^k.I[v-1]) - f_j.U[k] \cdot s'$$

ELSE $b_{left} := 0$;
2.1.2.13. IF $t_{right} < A$ THEN

$$b_{right} := \Psi_j^k.B[w+1] + \Psi_j^k.U[w+1] \cdot (t_{right} - \Psi_j^k.I[w]) - f_j.U[k] \cdot s'$$

ELSE $b_{right} := 0$;
2.1.2.14. IF $T' = \emptyset$ THEN $b_{inner} := 0$ ELSE

$$b_{inner} := \max_{s=v,v+1,\ldots,w}\{\Psi_j^k.B[s] + \Psi_j^k.U[s] \cdot (\Psi_j^k.I[s] - \Psi_j^k.I[s-1])\} - f_j.U[k] \cdot s';$$

2.1.2.15. Denote function

$$\varphi_{left}(x) := b_{left} - (f_j.U[k] - \Psi_j^k.U[v]) \cdot x.$$

IF $t_{left} = A$ THEN $\varphi_{left}(x) := 0$;

2.1.2.16. Denote function

$$\varphi_{right}(x) := b_{right} - (f_j.U[k] - \Psi_j^k.U[w+1]) \cdot x.$$

IF $t_{right} = A$ THEN $\varphi_{right}(x) := 0$;

2.1.2.17. Denote function

$$\varphi_{inner}(x) := b_{inner} - f_j.U[k] \cdot x.$$

IF $T' = \emptyset$ THEN $\varphi_{inner}(x) := 0$;

2.1.2.18. Construct the piecewise linear function

$$\varphi_{max}(x) := \max_{x \in [0,\varepsilon]} \{\varphi_{left}(x), \varphi_{right}(x), \varphi_{inner}(x)\}$$

according to Procedure *FindMax*;

2.1.2.19. add the values from $\varphi_{max}.I$ increased by s' to the set $\Phi_j^k.I$;

2.1.2.20. add the values from $\varphi_{max}.B$ to the set $\Phi_j^k.B$;

2.1.2.21. add the values from $\varphi_{max}.U$ to the set $\Phi_j^k.U$;

2.1.2.22. IF $\varepsilon = \varepsilon_{left}$ THEN exclude $\Psi_j^k.I[v]$ from the set T and $v := v + 1$;

2.1.2.23. IF $\varepsilon = \varepsilon_{right}$ THEN include $\Psi_j^k.I[w+1]$ to the set T and $w := w + 1$;

2.1.2.24. $s' := s' + \varepsilon$.

2.1.2.25. $t_{left} := s' + f_j.I[k-1]$, $t_{right} := \min\{s' + f_j.I[k], A\}$;

2.1.2.26. Modify the function Φ_j^k according to Procedure 2.2.

Lemma 2 *Procedure 2.1.2. has a running time of* $O(|F_{j-1}.I|)$.

Proof. Step [2.1.2.14] has to be performed with the use of a simple data structure. Let $\{q_1, q_2, \ldots, q_r\}$ be a maximal subset of T' having the following properties:

$q_1 < q_2 < \ldots < q_r$;
there is no $j \in T'$ such that $q_i \leq j < q_{i+1}$ and
$\Psi_j^i.B[j] \geq \Psi_j^i.B[q_{i+1}]$, $i = 1, \ldots, r-1$.

We can keep track of the set $\{q_1, q_2, \ldots, q_r\}$ by storing its elements in increasing order in a Queue Stack, i.e., a list with the property that elements at the beginning can only be deleted while at the end, elements can be deleted and added [2]. This data structure can easily be implemented such that each deletion and each addition requires a constant time. So, step [2.1.2.14] can be performed in constant time.

Each of the steps [2.1.2.6]–[2.1.2.25] can be performed in constant time. The loop [2.1.2.5.] can be performed in $O(|\Psi_j^k.I|)$ time, where $|\Psi_j^k.I| = |F_{j-1}(t).I|$, since each time a break point from $|\Psi_j^k.I|$ is added or deleted. So, the lemma is true. □

We remind that in the DPA, the functional equations (4) are considered. In fact, in Procedure 2.1.1., we construct the function

$$b_j^k - u_j^k t_j^{k-1} + u_j^k \cdot (t + x) + F_{j-1}(t + x)$$

and in Procedure 2.1.2., we construct the function

$$\Phi_j^k(t) = \max_{x \in (t_j^{k-1}, t_j^k] \bigcap [0, A-t]} \{b_j^k - u_j^k t_j^{k-1} + u_j^k \cdot (t+x) - u_j^k \cdot t + F_{j-1}(t+x)\}.$$

Unlike the DPA, to construct $\Phi_j^k(t)$ in the GrA, we do not consider all integer points $x \in (t_j^{k-1}, t_j^k] \bigcap [0, A-t]$, but only the break points from the interval, since only they influence the values of $\Phi_j^k(t)$ (and in addition t_{left}, t_{right}). Step [2.1.3.] can be performed according to Procedure FindMax as well, i.e., to construct $F_j(t) := \max\{F_j(t), \Phi_j^i(t)\}$, their linear fragments have to be compared in each interval, organized by their break points. It is easy to see that we do the same operation with the integer points t as in the DPA. So, the values $F_j(t),\ t \in Z$, are the same for the GrA and the DPA, and we can state the following:

Lemma 3 *The values $F_j(t),\ j = 1, 2, \ldots, n$, at the points $t \in [0, A] \bigcap Z$ are equal to the values of the functions $F_j(t)$ considered in the DPA.*

Procedure 2.2.
Given is $F_j(t)$;
FOR $k := 1$ TO $|F_j.I| - 1$ DO

IF $F_j.U[k] = F_j.U[k+1]$ AND $F_j.U[k] \cdot (F_j.U[k] - F_j.U[k-1]) + F_j.B[k] = F_j.B[k+1]$ THEN
$F_j.B[k+1] := F_j.B[k]$;
Delete the kth elements from $F_j.B,\ F_j.U$ and $F_j.I$;

So, in Procedure 2.2., we combine two adjoining linear fragments that are in the same line. That means that, if we have two adjacent linear fragments which are described by the values (slopes) $F_j.U[k],\ F_j.U[k+1]$ and $F_j.B[k], F_j.B[k+1]$, where

$$F_j.U[k] \cdot (F_j.U[k] - F_j.U[k-1]) + F_j.B[k] = F_j.B[k+1],$$

(i.e., these fragments are on the same line), then, to reduce the number of intervals $|F_j.I|$ and thus the running time of the algorithm, we can join these two intervals into one interval.

Lemma 4 *All functions $F_j^i(t),\ j = 1, 2, \ldots, n,\ i = 1, 2, \ldots, k_j$, are piecewise linear on the interval $[0, A]$ with integer break points.*

Proof. It is obvious that function $F_0(t)$ is piecewise linear on the interval $[0, A]$. In Procedure 2.1.1., all break points from the set $\Psi_1^i.I$ are integer as well (see the comments after Procedure 2.1.1.). Since all points from $f_1.I$ are integer, we have $\varepsilon \in Z$ and as a consequence, $s' \in Z$. According to the Procedure FindMax, all points $\varphi_{max}.I$ considered in Procedure 2.1.2. are integer. So, all break points from $\Phi_1^i.I,\ i = 1, 2, \ldots, k_1$, are integer as well. Thus, the break points of the function $F_1^i(t) := \max\{F_1^{i-1}(t), \Phi_1^i(t)\}$ are integer, if we use Procedure FindMax to compute the function $\max\{F_1^{i-1}(t), \Phi_1^i(t)\}$. Analogously, we can prove that all break points of $F_2^i(t),\ i = 1, 2, \ldots, k_2$ are integer, etc.

Thus, it is obvious that all functions $F_j^i(t),\ j = 1, 2, \ldots, n,\ i = 1, 2, \ldots, k_j$, constructed in the GrA are piecewise linear. □

Theorem 1 *The GrA finds an optimal solution of the problem in*

$$O\left(\sum k_j \min\{A, \max_{j=1,2,\ldots,n} \{|F_j.B|\}\}\right)$$

time.

Proof. Analogously to the proof of Lemma 4, after each step [2.1.3.] of the GrA, the function $F_j^i(t),\ j = 1,2,\ldots,n,\ i = 1,2,\ldots,k_j$, has only integer break points from the interval $[0, A]$. Each function $\Phi_j^i.I,\ j = 1,2,\ldots,n, i = 1,2,\ldots,k_j$, has only integer break points from $[0, A]$ as well. So, to perform step [2.1.3.], we need to perform Procedure FindMax on no more than $A + 1$ intervals. Thus, the running time of step [2.1.3.] is $O(A)$. According to Lemmas 1 and 2, the running time of steps [2.1.1.] and [2.1.2.] is $O(F_j^i.I)$, where $F_j^i.I \leq A$. The running time of step [2.2.] is $O(F_j^i.I)$ as well.

Analogously to the comments after the DPA, it is easy to show that $F_j^i(t),\ j = 1,2,\ldots,n$, is a non-increasing function in t. Thus,

$$F_j^i.B[k] \geq F_j^i.B[k+1],\ j = 1,2,\ldots,n,\ k = 1,2,\ldots,|F_j^i.I| - 1.$$

Then, according to Procedure 2.2., there are no more than $2 \cdot F_j^i.B[0]$ different values in the set $F_j^i.I$, where $F_j^i.B[0]$ is the maximal value in the set $F_j^i.B$.

Thus, the running time of the GrA is

$$O\left(\sum k_j \min\{A, \max_{j=1,2,\ldots,n,\ i=1,2,\ldots,k_j} \{|F_j^i.B|\}\}\right).$$

It is easy to show that this running time can be rewritten as

$$O\left(\sum k_j \min\{A, \max_{j=1,2,\ldots,n} \{|F_j.B|\}\}\right).$$

□

In [9], we illustrate the idea of the GrA by means of a numerical example in more detail.

4 An FPTAS based on the GrA

In this section, a fully polynomial-time approximation scheme (FPTAS) is derived based on the GrA presented in Section 3.

Let $LB = \max_{j=1,\ldots,n} f_j(A)$ be a lower bound and $UB = n \cdot LB$ be an upper bound on the optimal objective function value.

The idea of the FPTAS is as follows. Let $\delta = \frac{\varepsilon LB}{n}$. To reduce the time complexity of the GrA, we have to diminish the number of columns $|F_j.B|$ considered, which corresponds to the number of different objective function values $b \in F_j.B, b \leq UB$. If we do not consider the original values $b \in F_j.B$ but the values $\overline{b}$ which are rounded up or down to the nearest multiple of δ values b, there are no more than $\frac{UB}{\delta} = \frac{n^2}{\varepsilon}$ different values $\overline{b}$. Then we will be able to approximate the function $F_j(t)$ into a similar function with no more than $2\frac{n^2}{\varepsilon}$ break points. Furthermore, for such a modified table representing a function $\overline{F}_j(t)$, we will have

$$|F_j(t) - \overline{F_j}(t)| < \delta \leq \frac{\varepsilon F(\pi^*)}{n}.$$

If we do the rounding and modification after each step [2.2.], the cumulative error will be no more than $n\delta \leq \varepsilon F(\pi^*)$, and the total running time of the n runs of the step [2.2.] will be

$$O\left(\frac{n^2 \sum k_j}{\varepsilon}\right),$$

i.e., an FPTAS is obtained.

In [7], a technique was proposed to improve the complexity of an approximation algorithm for optimization problems. This technique can be described as follows. If there exists an FPTAS for a problem with a running time bounded by a polynomial $P(L, \frac{1}{\varepsilon}, \frac{UB}{LB})$, where L is the length of the problem instance and UB, LB are known upper and lower bounds, and the value $\frac{UB}{LB}$ is not bounded by a constant, then the technique enables us to find in $P(L, \log\log\frac{UB}{LB})$ time values UB_0 and LB_0 such that

$$LB_0 \leq F^* \leq UB_0 < 3LB_0,$$

i.e., $\frac{UB_0}{LB_0}$ is bounded by the constant 3. By using such values UB_0 and LB_0, the running time of the FPTAS will be reduced to $P(L, \frac{1}{\varepsilon})$, where P is the same polynomial. So, by using this technique, we can improve the FPTAS to have a running time of

$$O\left(\frac{n \cdot \sum k_j}{\varepsilon}(1 + \log\log n)\right).$$

A detailed description of an FPTAS based on a GrA for some single machine scheduling problems was presented in [6]. In the following table these GrA and FPTAS are summarized.

Table 1: Time complexity of the GrA and FPTAS based on the GrA

Problem	Time complexity of the GrA	Time complexity of the FPTAS	Time complexity of the classical DPA
$1\|\|\sum w_j U_j$	$O(\min\{2^n, n \cdot \min\{d_{max}, F_{opt}\}\})$	-	$O(nd_{max})$
$1\|d_j = d'_j + A\|\sum U_j$	$O(n^2)$	-	$O(n\sum p_j)$
$1\|\|\sum GT_j$	$O(\min\{2^n, n \cdot \{d_{max}, nF^*\}\})$	$O(n^2 \log\log n + \frac{n^2}{\varepsilon})$	$O(nd_{max})$
$1\|\|\sum T_j$ special case $B-1$	$O(\min\{2^n, n \cdot \min\{d_{max}, F^*\}\})$	$O(n^2/\varepsilon)$	$O(nd_{max})$
$1\|\|\sum T_j$ special case $B-1G$	$O(\min\{n^2 \cdot \min\{d_{max}, F^*\}\})$	$O(n^3/\varepsilon)$	$O(n^2 d_{max})$
$1\|d_j = d\|\sum w_j T_j$	$O(\min\{n^2 \cdot \min\{d, F^*\}\})$	$O(n^3/\varepsilon)$	$O(n^2 d_{max})$
$1(no\text{-}idle)\|\|\max\sum w_j T_j$	$O(\min\{2^n, n \cdot \min\{d_{max}, F^*, \sum w_j\}\})$	$O(n^2 \log\log n + \frac{n^2}{\varepsilon})$	$O(nd_{max})$
$1(no\text{-}idle)\|\|\max\sum T_j$	$O(n^2)$	-	$O(nd_{max})$

5 Concluding Remarks

In this paper, we used a graphical approach to improve a known pseudo-polynomial algorithm for the Investment Optimization Problem and to derive a FPTAS with the best known running time.

The graphical approach can be applied to problems, where a pseudo-polynomial algorithm exists and Boolean variables are used in the sense that yes/no decisions have to be made. However, for the knapsack problem, the graphical algorithm mostly reduces substantially the number of states to be considered but the time complexity of the algorithm remains pseudo-polynomial. For example, for the single machine problem of maximizing total tardiness, the complexity of the graphical algorithm is $O(n^2)$, while the complexity of the dynamic programming algorithm is $O(n \sum p_j)$. Thus, the graphical approach is not only of a practical but also of a theoretical importance.

References

1. A.A. Lazarev and F. Werner, A Graphical Realization of the Dynamic Programming Method for Solving NP-hard Problems. Computers & Mathematics with Applications. Vol. 58, No. 4, 2009, 619 – 631.
2. A.V. Aho, J.E. Hopcroft and J.D. Ullman, Data Structures and Algorithms, Addison-Wesley, London, 1983.
3. H. Kellerer, U. Pferschy and D. Pisinger, Knapsack Problems, Springer-Verlag, Berlin, 2004.
4. D.X. Shaw and A. P. M. Wagelmans, An Algorithm for Single-Item Capacitated Economic Lot Sizing with Piecewise Linear Production Costs and General Holding Costs, Management Science, Vol. 44, No. 6, 1998, 831–838.
5. S. Kameshwaran and Y. Narahari, Nonconvex Piecewise Linear Knapsack Problems, European Journal of Operational Research, 192, 2009, 56–68.
6. E.R. Gafarov, A. Dolgui and F. Werner F.: Dynamic Programming Approach to Design FPTAS for Single Machine Scheduling Problems, Research Report LIMOS UMR CNRS 6158, 2012.
7. S. Chubanov, M.Y. Kovalyov and E. Pesch, An FPTAS for a Single-Item Capacitated Economic Lot-Sizing Problem with Monotone Cost Structure, Math. Program., Ser. A 106, 2006, 453 – 466.
8. K. Schmelev, X. Delorme, A. Dolgui, F. Grimaud and M.Y. Kovalev: Lot-Sizing on a Single Machine, ILP Models (submitted in May 2012).
9. E.R. Gafarov, A.A. Lazarev, A. Dolgui and F. Werner, An Improved Graphical Approach for an Investment Optimization Problem: Algorithm and a Numerical Example, Preprint 02/13, FMA, Otto-von-Guericke-Universität Magdeburg, 2013, 20 pages.