A FPTAS for Minimizing Positive Half-Product

Adam Janiak
Institute of Engineering Cybernetics, Wroclaw University of Technology, Janiszewskiego 11/17, 50-372 Wroclaw, Poland.
janiak@ict.pwr.wroc.pl

Mikhail Y. Kovalyov
Institute of Engineering Cybernetics, National Academy of Sciences of Belarus, Surganova 6, 220012 Minsk, Belarus.
koval@newman.bas-net.by

Wieslaw Kubiak
Faculty of Business Administration, Memorial University of Newfoundland, A1B 3X5 St. John's, Canada.
wkubiak@mun.ca

Frank Werner
Otto-von-Guericke University of Magdeburg, Postfach 4120, 39016 Magdeburg, Germany.
frank.werner@mathematik.uni-magdeburg.de

Abstract

The problem of minimizing a quadratic function of boolean variables, which we call PHP (positive half-product), is formulated. A fully polynomial time approximation scheme (FPTAS) for PHP is derived. Several NP-hard scheduling problems can be formulated in terms of PHP. In this presentation, we concentrate on a single machine scheduling problem with controllable job processing times to minimize a linear combination of the total weighted job completion time and the total weighted processing time compression.

1. Problems HP (half-product) and PHP (positive half-product)

Denote $x = (x_1, \ldots, x_n)$. The problem of minimizing the half-product, which we denote as HP, can be formulated as follows:

$$HP(x) = \sum_{1 \le i < j \le n} a_i b_j x_i x_j - \sum_{j=1}^{n} h_j x_j \to \min,$$

$$\text{subject to } x_j \in \{0,1\}, a_j, b_j, h_j \ge 0,\ j = 1, \ldots, n.$$

Badics and Boros [1] have proved that problem HP is NP-hard by using a reduction from PARTITION. They also derived a FPTAS for problem HP.

Let us remind a definition of a FPTAS for a minimization problem. Denote by F* and F^0 the values of an optimal and approximate solutions for this problem, respectively. Given $e > 0$, an approximate solution is called an *e-approximate* one, if $F^0 - F^* \le e|F^*|$. A family of approximation algorithms $\{G_e\}$ forms a FPTAS if G_e delivers an *e*-approximate solution in time polynomial in the problem input length in binary encoding and $1/e$.

There is an equivalence between problem HP and some scheduling problems in a sense that their exact solutions coincide. However, *e*-approximate solution to problem HP is not an *e*-approximate solution to a scheduling problem in most of the cases.

We now formulate a problem which we call PHP (*positive half-product*). Several scheduling problems can be formulated in terms of problem PHP and exact and *e*-approximate solutions of problem PHP and these scheduling problems coincide.

Problem PHP:

$$F(x) = \sum_{1\le i<j\le n} a_i b_j x_i x_j + \sum_{j=1}^{n} h_j(1-x_j) - \sum_{j=1}^{n} g_j x_j + D \to \min,$$

$$\text{subject to } x_j \in \{0,1\}, a_j, b_j, h_j, g_j, D \ge 0,\ j = 1,\ldots,n.$$

Let $x^* = (x_1^*,\ldots,x_n^*)$ be an optimal solution to this problem.

2. A FPTAS for problem PHP

Our FPTAS is based on a combination of the ideas given by De, Ghosh and Wells [2], Badics and Boros [1] and Kovalyov and Kubiak [5]. We need the following statement which shows how to represent objective function $F(x_1,\ldots,x_n)$ as a combination of several auxiliary functions such that some of them depend on $x_1,\ldots,x_k$ only and the others depend on $x_{k+1},\ldots,x_n$ only.

Lemma 1 For any 0-1 vector *x*, we have

$$F(x) = F_{1,k}(x) + a_{1,k}(x)b_{k+1,n}(x) + F_{k+1,n}(x),$$

where

$$F_{1,k}(x) = \sum_{1\le i<j\le k} a_i b_j x_i x_j + \sum_{j=1}^{k} h_j(1-x_j) + \sum_{j=1}^{k} g_j x_j,$$

$$F_{k+1,n}(x) = \sum_{k+1\le i<j\le n} a_i b_j x_i x_j + \sum_{j=k+1}^{n} h_j(1-x_j) + \sum_{j=k+1}^{n} g_j x_j + D,$$

$$a_{1,k}(x) = \sum_{j=1}^{k} a_j x_j,\quad b_{k+1,n}(x) = \sum_{j=k+1}^{n} b_j x_j.$$

Now we are able to describe our FPTAS which we denote as $\{G_e\}$. It is a standard FPTAS based on trimming-of-the-state-space technique of Ibarra and Kim [3]. In the algorithm G_e sets $X_1,\ldots,X_n$ of partial solutions to problem PHP are recursively constructed, where X_k is a subset of 0-1 vectors $x = (x_1,\ldots,x_k)$. Partial solutions are filtered so that in each set X_k there is a solution $(x_1,\ldots,x_k)$ such that

$$\left|F(x_1,\ldots,x_k,x_{k+1}^*,\ldots,x_n^*) - F(x^*)\right| \le eF(x^*).$$

Due to this filtering, an *e*-approximate solution is guaranteed to be contained in the set X_n.

Algorithm G_e

Step 1. **(Initialization)** Determine $d > 0$ such that $(1+d)^n = 1+e$. Set $j = 0$ and $X_0 = \{(f)\}$.

Step 2. **(Recursion)** Construct set $Y_k = \{(x,0),(x,1) \mid x \in X_{k-1}\}$. If $k = n$, then set $X_n = Y_n$ and go to Step 3. Partition set Y_k into subsets $Y_{r,k}$, $r = 1,\ldots,s_k$, such that

$$\left|a_{1,k}(x') - a_{1,k}(x'')\right| \le d\ \min\{a_{1,k}(x'), a_{1,k}(x'')\}$$

for any x' and x'' from the same subset. In each subset $Y_{r,k}$, choose a vector $x^{r,k}$ such that $F_{1,k}(x^{r,k}) = \min\{F_{1,k}(x^{r,k}) \mid x \in Y_{r,k}\}$. Set $X_k = \{x^{r,k} \mid r = 1,\ldots,s_k\}$, $k = k+1$ and repeat Step 2.

Step 3. **(ε-approximate solution)** Choose an *e*-approximate solution $x^e \in X_n$ such that $F(x^e) = \min\{F(x) \mid x \in X_n\}$ and stop.

Theorem 1 Algorithm G_e constructs a 0-1 vector $x^e \in X_n$ such that $F(x^e) - F(x^*) \le eF(x^*)$ and does it in $O(n^2 \log A/e)$ time, where $A = \sum_{j=1}^{n} a_j$.

3. Single machine scheduling with controllable processing times

There are *n* independent and non-preemptive jobs to be scheduled for processing on a single machine. All jobs are available for processing at time zero. The processing time of job *j* is a variable $p_j \in [l_j, u_j]$. A solution is represented by an ordered pair (p, p), where ð is a sequence of jobs and $p = (p_1,\ldots,p_n)$ is a vector of the job processing times. Given a feasible pair (p, p), its quality is evaluated by the combined objective function $TWC = \sum_{j=1}^{n} w_j C_j + \sum_{j=1}^{n} v_j (u_j - p_j)$ to be minimized. Here C_j is the completion time of job *j*. All numerical parameters are assumed to be positive integers. The values of variables p_j, $j = 1,\ldots,n$, are assumed to be non-negative real numbers. We denote the above problem as TWC.

Observe that, for problem TWC, a search for optimal job processing times can be limited as follows.

Lemma 2 There exists an optimal solution to problem TWC, in which $p_j \in \{l_j, u_j\}$, $j = 1,\ldots,n$.

A search for an optimal job sequence can be limited based on the following lemma.

Lemma 3 There exists an optimal solution to problem TWC, in which jobs with processing times $p_j = l_j$ are sequenced in the non-decreasing order of the ratio l_j / w_j and jobs with processing times $p_j = u_j$ are sequenced in the non-decreasing order of the ratio u_j / w_j .

In this presentation, we consider the case when $l_j = 0 \quad j = 1,\ldots,n$. We denote this case as TWC_0.

Renumber jobs so that $u_1 / w_1 \le \ldots \le u_n / w_n$. Since $l_j = 0 \quad j = 1,\ldots,n$, and taking into account Lemmas 1 and 2, problem TWC_0 reduces to finding a partition of the set of jobs into two subsets: a subset of jobs with $p_j = 0$ a subset of jobs with $p_j = u_j$. In an optimal schedule, jobs of the first subset are processed first in arbitrary order followed by the jobs of the second subset in the increasing order of their indices.

Theorem 2 Problem TWC_0 and problem HP are equivalent.

Proof. Consider an arbitrary optimal solution to problem TWC_0, which satisfies Lemmas 1 and 2. Introduce 0-1 variables x_j such that $x_j = 1$ if $p_j = u_j$ and $x_j = 0$ if $p_j = 0$. Given $(x_1,\ldots,x_n)$ calculate objective function value for problem TWC_0:

$$TWC_0(x) = \sum_{j=1}^{n} w_j x_j \sum_{i=1}^{j} u_i x_i + \sum_{j=1}^{n} v_j u_j (1 - x_j) = \sum_{1 \le i < j \le n} u_i w_j x_i x_j - \sum_{j=1}^{n} u_j (v_j - w_j) x_j + \sum_{j=1}^{n} v_j u_j .$$

It is easy to see that $TWC_0(x) = HP(x) + D$ if $a_j = u_j$, $b_j = w_j$, $h_j = u_j(v_j - w_j)$, $j = 1,\ldots,n$ and $D = \sum_{j=1}^{n} v_j u_j = \sum_{j=1}^{n} (h_j + a_j b_j)$. Therefore, vector x which is optimal for problem TWC_0 is optimal for the corresponding problem HP and vice versa.

Since Badics and Boros [1] have proved that problem HP is NP-hard, problem TWC_0 is NP-hard as well.

Problem HP admits two dynamic programming formulations and corresponding algorithms which are similar to those derived by Jurish, Kubiak and Jozefowska [4] for a problem of finding a minimum weighted clique in a complete graph. Set $B = \sum_{j=1}^{n} b_j$ and $A = \sum_{j=1}^{n} a_j$. One of the algorithms runs in $O(nB)$ time, thus solving problem TWC_0 with unit weights w_j in $O(n^2)$ time. The other algorithm runs in $O(nA)$ time and, therefore, solves problem TWC_0 with unit weights u_j in $O(n^2)$ time.

A FPTAS of Badics and Boros for problem HP cannot be directly used for problem TWC_0.

Theorem 3 Problem TWC_0 and problem PHP coincide.

Proof. We have $TWC_0(x) = F(x)$ by setting $D = 0$, $a_j = u_j, b_j = w_j, h_j = u_j v_j$ and $g_j = u_j w_j$, $j = 1,\ldots,n$. Therefore, problems TWC_0 and PHP are, in fact, the same problems.

This theorem shows that the derived FPTAS for problem PHP is a FPTAS for problem TWC_0.

References

[1] T. Badics and E. Boros, Minimization of half-products, Mathematics of Operations Research 23 (1998), 649-660.

[2] P. De, J.B. Ghosh and C.E. Wells, On the minimization of completion time variance with bicriteria extention, Operations Research 40 (1992) 1148-1155.

[3] O. Ibarra and C.E. Kim, Fast approximation algorithms for the knapsack and sum of subset problems, Journal of the ACM 22 (1975) 463-468.

[4] B. Jurisch, W. Kubiak and J. Jozefowska, Algorithms for minclique scheduling problems. Discrete Applied Mathematics 72 (1997), 115-139.

[5] M.Y. Kovalyov and W. Kubiak, Fully polynomial approximation schemes for decomposable partition problems, Operations Research Proceedings 1999, Selected papers of the Symposium on Operations Research (SOR'99), Magdeburg, September 1-3, 1999, 397-401.) 273-280.