Advances in Difference Equations
a SpringerOpen Journal

RESEARCH Open Access

Triangle structure diagrams for a single machine batching problem with identical jobs

Shanlin Li[1*], Maoqin Li[1] and Hong Yan[2]

*Correspondence:
lishanlin56@hotmail.com
[1]Department of Mathematics, Taizhou University, Taizhou, Zhejiang 317000, P.R. China
Full list of author information is available at the end of the article

Abstract
The problem of batching identical jobs on a single machine to minimize the completion time is studied by employing the difference analysis technique. Constant processing times and batch setup times are assumed. We first establish the relation between the optimal solution and the first-order difference of the optimal objective function in terms of the number of jobs and investigate the properties of the first-order difference. Then we obtain the triangle structure diagram of the batching problem in $O(\sqrt{n})$ time at most by using the permutation of some numbers which describe the character of the first-order difference. The diagrams enable us to see clearly the specific expressions of optimal solutions for the n-jobs batching problem and any m-jobs batching problem simultaneously, where $m \leq n$. Also, we show that the result proposed by Santos (MSc thesis, 1984) and Santos and Magazine (Oper. Res. Lett. 4:99-103, 1985) is a special case of our result.
MSC: 90B30; 90B35

Keywords: scheduling; batching; triangle structure; single machine

1 Introduction

Consider the problem of scheduling identical jobs on a single machine, in which the jobs are processed in batches, with a setup time for each batch. The completion time of a job coincides with the completion time of the last scheduled job in its batch and all jobs in this batch have the same completion time. For a given number of jobs, we want to choose batch sizes so as to minimize the sum of the completion times of the jobs. There is a trade-off between keeping the number of setups incurred small, by having large batches, and keeping small the time each job waits for its batch to finish, by having small batches.

Formally, there is a set of n jobs with identical processing time, $J = \{j_1, \ldots, j_n\} = \{1, \ldots, n\}$, to be processed on a single machine. In a given schedule, for each job $j \in J$, we denote by C_j its completion time. The problem is, given job processing time p and setup time S, to find the number of batches k, and batch sizes b_i, such that $\sum_{i=1}^{k} b_i = n$, so as to minimize $\sum_{j=1}^{n} C_j = \sum_{i=1}^{k} b_i \sum_{j=1}^{i} (S + b_j p)$. The problem is referred to as $1|p_j = p, S - batch| \sum C_j$.

There are several different versions of solving the problem $1|p_j = p, S - batch| \sum C_j$. Coffman *et al.* [1] propose a backward dynamic programming algorithm, running in $O(n^2)$ time. A faster solution algorithm is given Naddef and Santos [2], running in $O((p/s)n)$ time. Another is given by Coffman, Nozari and Yannakakis [3], running in $O(\sqrt{n})$ time. Shallcross [4] presents an optimal solution of the form $b_i = \begin{cases} \lfloor (c - iS)/p \rfloor, & 1 \leq i \leq l, \\ \lfloor (c - iS - 1)/p \rfloor, & l < i \leq k, \end{cases}$ where c

and l can be determined by the algorithm running in $O(\log p \log(np))$. However, Potts and Kovalyov [5] point out that the algorithm by Shallcross [4] is rather intricate. All of the optimal solutions mentioned above are not of the specific expression. Potts and Kovalyov [5] point out that a key to the development of a polynomial algorithm is provided by Santos [6] and Santos and Magazine [7] who analyze a continuous relaxation in which batch sizes are not constrained to be integer. Specifically, they show that the optimal number of batches is $k = \lceil\sqrt{1/4 + 2np/S} - 1/2\rceil$ and the optimal batch sizes b_i are $n/k + S(k+1)/2p - iS/p$ for $i = 1, \ldots, k$.

The results by Santos [6] and Santos and Magazine [7] are important because the specific expressions of the optimal solutions are given, which contributes so as not only to be much easier to implement in solving the problem than other algorithms, but also it entails a proof of the complexity for other scheduling problems. For example, Albers and Brucker [8] give the NP-hardness proofs of two batching problems by using the expression by Santos [6] and Santos and Magazine [7]. One slightly regrets that the specific expressions of the optimal solutions are only appropriate for some of the problem instances. One would like to know whether there is a specific solution formula that applies to all of the problem instances. This is the reason for studying the problem here. In this paper, we first establish the relation between the optimal solution and the first-order difference of the optimal objective function in terms of the number of jobs and we investigate the properties of the first-order difference. Then, we obtain the triangle structure diagram of the batching problem in $O(\sqrt{n})$ time at most by using the permutation of some numbers which describe the character of the first-order difference. The diagrams enable us to see clearly the specific expressions of optimal solutions for the n-jobs batching problem and any m-jobs batching problem simultaneously, where $m \leq n$.

This paper is organized as follows. In Section 2 we build the relation between the optimal solution and the first-order difference of the optimal objective function in terms of the number of jobs and investigate the properties of the first-order difference. The triangle structure diagrams are shown in the case $S = vp$ in Section 3 and in the case $S = vp + \varepsilon$ in Section 4, respectively, where v and $0 < \varepsilon < p$ are integers. Section 5 contains a conclusion and a discussion of some possible extensions.

2 Properties of the first-order difference

In this section, we introduce the concept of differences of the optimal objective function in terms of the number of jobs, and we show that the first-order difference sequence is strictly monotone increasing and each term in the second-order difference sequence, that is, each increment of the first-order difference terms, is between p and $2p$. Using the properties, we can reduce the batch sizing problem to finding some interval such that some fixed number exactly fall in it, where the interval consists of two adjacent terms in the first-order difference sequence.

For ease of presentation, we sequence the jobs according to nonincreasing indices. Any solution of problem $1|p_j = p, S - batch|\sum C_j$ is of the form

$$BS : Sn_k \cdots (n_{k-1}+1)Sn_{k-1} \cdots (n_{k-2}+1) \cdots Sn_1 \cdots 1,$$

where k is the number of batches,

$$1 \leq n_1 < n_2 < \cdots < n_k = n$$

and the batch sizes are $b_1 = n_k - n_{k-1}, \dots, b_{k-1} = n_2 - n_1, b_k = n_1$. Every solution BS corresponds to an objective function value

$$F(BS) = \sum_{j=1}^{n} C_j = \sum_{i=1}^{k} b_i \sum_{j=1}^{i} (S + b_j p) = \sum_{i=1}^{k} n_i \bigl(S + (n_i - n_{i-1})p\bigr),$$

where $n_0 = 0$.

In order to solve the batch sizing problem, we obviously have to find a constant k and a sequence of indices

$$1 \le n_1 < n_2 < \cdots < n_k = n$$

such that the above objective function value is minimized. Clearly, problem $1|p_j = p, S - batch|\sum C_j$ is a trivial matter when $S \le p$. We have the following result.

Theorem 1 *If $S \le p$, then for problem $1|p_j = p, S - batch|\sum C_j$ there exists an optimal solution in which $k = n$ and $n_i = i$ for $i = 1, \dots, n$, that is, $b_1 = \cdots = b_k = 1$.*

By Theorem 1, we assume that $S > p$ hereafter.

Let $F(j)$ denote the minimum the sum of the completion times for the j-jobs batching problem containing jobs $1, \dots, j$. Coffman *et al.* [1] propose a backward dynamic programming algorithm. The initialization is

$$F(0) = 0$$

and the recursion for $j = 1, \dots, n$ is

$$F(j) = \min\bigl\{j\bigl[S + (j - i)p\bigr] + F(i) \mid 0 \le i \le j - 1\bigr\}.$$

Under the most natural implementation, the algorithm requires $O(n^2)$ time. Below we do further research into the recursion formula, so as to determine completely the optimal successor t of j, that is, an integer t with $0 \le t \le j - 1$ such that

$$F(j) = j\bigl[S + (j - t)p\bigr] + F(t) \tag{1}$$

holds.

For $0 \le i \le j - 1$, set

$$F(j, i) = j\bigl[S + (j - i)p\bigr] + F(i).$$

Thus, we have

$$F(j) = \min_{i=1}^{j-1} F(j, i).$$

Let $\{\Delta_1(i)\}_{i=0}^{+\infty}$ denote the first-order difference sequence of the optimal objective function in terms of the number of jobs, where $\Delta_1(i) = F(i + 1) - F(i)$ for $i = 0, 1, 2, \dots$. The relation $F(j, i+1) < (>) F(j, i)$ stating that $i+1$ is better (worse) than i as a successor of j is equivalent to

$$\Delta_1(i) = F(i+1) - F(i) < (>) jp.$$

To break ties when choosing the successor, we assume that if $\Delta_1(i) = F(i+1) - F(i) = jp$, that is, $F(j, i+1) = F(j, i)$, then $i+1$ is better than i as a successor of j. Thus, if the sequence $\{\Delta_1(i)\}_{i=0}^{+\infty}$ is strictly monotone increasing, the determining optimal successor of j problem can be reduced to finding a nonnegative integer t such that $\Delta_1(t-1) \le jp < \Delta_1(t)$, where we define $\Delta_1(-1) = 0$. We denote by $SUCC(j)$ the optimal successor of j hereafter.

Proposition 1 *Assume that the sequence* $\{\Delta_1(i)\}_{i=-1}^{+\infty}$ *is strictly monotone increasing and* $\Delta_1(t) - \Delta_1(t-1) \ge p$ *for* $t = 1, 2, \ldots$. *Then for each j there is a unique* t_0 *with* $0 \le t_0 \le j-1$ *such that* $\Delta_1(t_0 - 1) \le jp < \Delta_1(t_0)$ *and* $t_0 = SUCC(j)$.

Proof We first prove the existence by induction. When $j = 1$, we have $0 = \Delta_1(-1) \le p < \Delta_1(0) = F(1) - F(0) = S + p$ and $SUCC(1) = 0$. Now suppose that it holds when $j \le h$ for some positive integer h, *i.e.* there is t_0 with $0 \le t_0 \le h-1$ such that $\Delta_1(t_0 - 1) \le hp < \Delta_1(t_0) < \cdots < \Delta_1(h-1)$. For $j = h+1$, by the proposition and induction assumptions, we have $0 = \Delta_1(-1) \le (h+1)p = hp + p < \Delta_1(h-1) + p \le \Delta_1(h)$. Thus, the existence holds for $j = h+1$, and it follows that $(\Delta_1(-1), \Delta_1(0), \ldots, \Delta_1(h))$ is a partition of $[0, \Delta_1(h))$.

Due to the fact that $(\Delta_1(-1), \Delta_1(0), \ldots, \Delta_1(j-1))$ is a partition of $[0, \Delta_1(j-1))$ again, the uniqueness holds.

Since the inequalities $\Delta_1(-1) < \Delta_1(0) < \cdots < \Delta_1(t_0 - 1) \le jp$ imply that t_0 is better than $t_0 - 1$, $t_0 - 1$ better than $t_0 - 2, \ldots, 1$ better than 0 in order and the inequalities $jp < \Delta_1(t_0) < \cdots < \Delta_1(j-1)$ imply that t_0 is better than $t_0 + 1$, $t_0 + 1$ better than $t_0 + 2, \ldots, j-2$ better than $j-1$ in order as a successor of j, we have $t_0 = SUCC(j)$. □

Now we give the key properties of the first-order difference.

Proposition 2 *If* $S > p$, *then the sequence* $\{\Delta_1(i)\}_{i=-1}^{+\infty}$ *is strictly monotone increasing and* $p \le \Delta_1(t) - \Delta_1(t-1) \le 2p$ *for* $t = 1, 2, \ldots$.

Proof We prove this proposition by induction. Simple calculations yield

$$\Delta_1(0) = F(1) - F(0) = S + p,$$

$$F(2) = \min\{F(2,1), F(2,0)\} = \min\{2(S+p) + S + p, 2(S+2p)\} = 2(S+2p),$$

$$\Delta_1(1) = F(2) - F(1) = S + 3p > \Delta_1(0) > \Delta_1(-1) = 0$$

and

$$\Delta_1(1) - \Delta_1(0) = 2p.$$

Now suppose that it holds when $j \le h$ for some positive integer h, *i.e.* $\Delta_1(-1) < \Delta_1(0) < \cdots < \Delta_1(h)$ and $p \le \Delta_1(t) - \Delta_1(t-1) \le 2p$ for $t = 1, 2, \ldots, h$. We need to show that it holds when $j = h+1$.

By the induction assumption and Proposition 1, we may assume that $SUCC(h+1) = t_0$, where $0 \le t_0 \le h$ and $\Delta_1(t_0-1) \le (h+1)p < \Delta_1(t_0)$. Since $p \le \Delta_1(t_0) - \Delta_1(t_0-1)$ and $2p < S + p = \Delta_1(0) - \Delta_1(-1)$, we see that the optimal successor of h is either $t_0 - 1$ or t_0 and $0 \le t_0 - 1 \le h$. Noting the fact that $2p < S + p = \Delta_1(0) - \Delta_1(-1)$ and $\Delta_1(1) - \Delta_1(0) = 2p$, along with $p \le \Delta_1(t) - \Delta_1(t-1)$ for $t = 2, \dots, h$, we have $\Delta_1(h) > (h+2)p$. Thus, the optimal successor of $h+2$ is between 0 and h. By similar arguments for h, we see that the optimal successor of $h+2$ is either t_0 or t_0+1 and $0 \le t_0 + 1 \le h$. There are four cases to consider: (a) $SUCC(h) = SUCC(h+1) = SUCC(h+2) = t_0$, (b) $SUCC(h) = t_0 - 1$, $SUCC(h+1) = SUCC(h+2) = t_0$, (c) $SUCC(h) = SUCC(h+1) = t_0$, $SUCC(h+2) = t_0 + 1$, (d) $SUCC(h) = t_0 - 1$, $SUCC(h+1) = t_0$, $SUCC(h+2) = t_0 + 1$.

Case (a) $SUCC(h) = SUCC(h+1) = SUCC(h+2) = t_0$. In this case,

$$\begin{aligned}
&F(h) = h\bigl[S + (h - t_0)p\bigr] + F(t_0),\\
&F(h+1) = (h+1)\bigl[S + (h+1-t_0)p\bigr] + F(t_0),\\
&F(h+2) = (h+2)\bigl[S + (h+2-t_0)p\bigr] + F(t_0).
\end{aligned}$$

Thus, we have

$$\Delta_1(h+1) - \Delta_1(h) = 2p. \tag{2}$$

Case (b) $SUCC(h) = t_0 - 1$, $SUCC(h+1) = SUCC(h+2) = t_0$. In this case,

$$\begin{aligned}
&F(h) = h\bigl[S + (h - t_0 + 1)p\bigr] + F(t_0 - 1),\\
&F(h+1) = (h+1)\bigl[S + (h+1-t_0)p\bigr] + F(t_0),\\
&F(h+2) = (h+2)\bigl[S + (h+2-t_0)p\bigr] + F(t_0).
\end{aligned}$$

Thus, we have

$$\Delta_1(h+1) - \Delta_1(h) = (h+2)p - \Delta_1(t_0 - 1). \tag{3}$$

Since $SUCC(h+1) = SUCC(h+2) = t_0$, the inequalities $\Delta_1(t_0) > (h+2)p > (h+1)p \ge \Delta_1(t_0 - 1)$ hold. This implies $\Delta_1(h+1) - \Delta_1(h) = (h+2)p - \Delta_1(t_0-1) \ge p$. Noting the fact that $1 \le t_0 \le h$, by the induction assumption, we have $\Delta_1(h+1) - \Delta_1(h) = (h+2)p - \Delta_1(t_0-1) \le \Delta_1(t_0) - \Delta_1(t_0-1) \le 2p$.

Case (c) $SUCC(h) = SUCC(h+1) = t_0$, $SUCC(h+2) = t_0 + 1$. In this case,

$$\begin{aligned}
&F(h) = h\bigl[S + (h - t_0)p\bigr] + F(t_0),\\
&F(h+1) = (h+1)\bigl[S + (h+1-t_0)p\bigr] + F(t_0),\\
&F(h+2) = (h+2)\bigl[S + (h+2-t_0-1)p\bigr] + F(t_0+1).
\end{aligned}$$

Thus, we have

$$\Delta_1(h+1) - \Delta_1(h) = \Delta_1(t_0) - hp. \tag{4}$$

Since $SUCC(h) = SUCC(h+1) = t_0$, the inequalities $\Delta_1(t_0) > (h+1)p > hp$ hold. This implies $\Delta_1(h+1) - \Delta_1(h) = \Delta_1(t_0) - hp \geq p$. Also, we have $\Delta_1(h+1) - \Delta_1(h) = \Delta_1(t_0) - hp \leq 2p$. Otherwise, $\Delta_1(t_0) - hp > 2p$ implies $\Delta_1(t_0) > (h+2)p$. This contradicts with the fact that $\Delta_1(t_0) \leq (h+2)p < \Delta_1(t_0+1)$.

Case (d) $SUCC(h) = t_0 - 1$, $SUCC(h+1) = t_0$, $SUCC(h+2) = t_0 + 1$. In this case,

$$\begin{aligned}
&F(h) = h\bigl[S + (h+1-t_0)p\bigr] + F(t_0 - 1),\\
&F(h+1) = (h+1)\bigl[S + (h+1-t_0)p\bigr] + F(t_0),\\
&F(h+2) = (h+2)\bigl[S + (h+1-t_0)p\bigr] + F(t_0 + 1).
\end{aligned}$$

Thus, we have

$$\Delta_1(h+1) - \Delta_1(h) = \Delta_1(t_0) - \Delta_1(t_0 - 1). \tag{5}$$

Since $1 \leq t_0 \leq h$, along with the induction assumption, the inequalities $p \leq \Delta_1(h+1) - \Delta_1(h) \leq 2p$ hold. □

These two propositions, as well as the four formulas yielded in the proof of Proposition 2, are very important for our batch sizing problem. Proposition 1 ensures that the problem can be reduced to finding the interval formed by the first-order difference problem. Proposition 2 and the four formulas make it possible to determine exactly the first-order difference sequence. $p \leq \Delta_1(h+1) - \Delta_1(h) \leq 2p$ implies that each nonnegative integer must be the optimal successor of one or two positive integers, and by the monotonicity each positive integer has a unique nonnegative integer as its optimal successor. Based on the successor membership between these integers, we can obtain a partition of the set of positive integers. We define $N_k = \{j \mid$ the optimal number of batches of j-jobs batching problem is equal to $k\}$ for $k = 1, 2, \ldots,$ called the k-batches case set of numbers of jobs, and the sequence of integers in N_k is in increasing natural order. Then $(N_1, \ldots, N_k, \ldots)$ is a partition of the set of position integers. Let $C_{1,i} = i$ for $i = 1, \ldots, |N_1|$. Then $N_1 = (C_{1,1}, \ldots, C_{1,|N_1|})$. We define $C_{k,i} = \{j \mid SUCC(j) \in C_{k-1,i}\}$ for $k = 2, 3, \ldots$ and $i = 1, \ldots, |N_1|$, called the ith periodic set of N_k, and the sequence of integers in $C_{k,i}$ is in increasing natural order. Then $N_k = (C_{k,1}, \ldots, C_{k,|N_1|})$ and $(C_{1,1}, \ldots, C_{1,|N_1|}, \ldots, C_{k,1}, \ldots, C_{k,|N_1|}, \ldots)$ is a partition of the set of position integers. Based on the successor membership between these periodic sets, we now give a basic diagram of the batching problem as follows, where the nodes in the diagram consist of the integers in C_{ki} (see Figure 1).

In the following sections, we determine exactly the number of integers in $C_{k,i}$ and the optimal successor membership between the integers.

Figure 1 A basic diagram of the batching problem.

$$\begin{array}{ccc}
C_{1,1} & \cdots & C_{1,|N_1|}\\
C_{2,1} & \cdots & C_{2,|N_1|}\\
\cdots & \cdots & \cdots\\
C_{k,1} & \cdots & C_{k,|N_1|}\\
\cdots & \cdots & \cdots
\end{array}$$

3 The triangle structure diagram in the case $S = vp$

In this section we present a specific diagram of the batching problem with $S = vp$, which can be obtained in constant time. Since the diagram consists of v triangles in the shape, we call it a triangle structure diagram. Using the optimal successor membership between the integers presented by the diagram, we give the specific expression of the optimal solution for an any-number-jobs batching problem with $S = vp$, and show that the result proposed by Santos [6] and Santos and Magazine [7] is a special case of our result.

The diagram depends exactly on the first-order difference sequence. To compute the sequence, we can first obtain values $\Delta_1(0), \Delta_1(1), \ldots, \Delta_1(|N_1|)$ by simple calculations. Then using the results of the previous section and the special properties of the first-order difference in the case $S = vp$, which will be given below, we can determine the first-order difference of the integers in N_2. Generally, we can compute the first-order difference of the integers in N_k if those in N_{k-1} are already known. The following two propositions describe the special properties of the first-order difference in the case $S = vp$.

Proposition 3 *Assume $S = vp$. Then $\Delta_1(1) - \Delta_1(0) = 2p$, $\Delta_1(j) - \Delta_1(j-1) = 2p$ if j and $j-1$ have the same optimal successor and $\Delta_1(j) - \Delta_1(j-1) = p$ otherwise for $j = 2, 3, \ldots$.*

Proof Since $\Delta_1(0) = F(1) - F(0) = S + p$ and $\Delta_1(1) = F(2) - F(1) = S + 3p$, we have $\Delta_1(1) - \Delta_1(0) = 2p$. Suppose $SUCC(j) = t_0$, where $t_0 \geq 0$.

When $SUCC(j-1) = SUCC(j) = SUCC(j+1) = t_0$, by (2), we have $\Delta_1(j) - \Delta_1(j-1) = 2p$.

When $SUCC(j-1) = SUCC(j) = t_0$ and $SUCC(j+1) = t_0 + 1$, it follows from Proposition 1 that

$$\Delta_1(t_0 - 1) \leq (j-1)p < jp < \Delta_1(t_0) \leq (j+1)p. \tag{6}$$

Due to $S = vp$ and $F(i) = i(S + (i-t)p) + F(t)$, where $SUCC(i) = t$, $F(i)/p$ is an integer for $i = 1, 2, \ldots$. Thus, $\Delta_1(i)/p$ is an integer for $i = 0, 1, 2, \ldots$. By equation (6), $\Delta_1(t_0) - (j-1)p = 2p$. So, we have $\Delta_1(j) - \Delta_1(j-1) = \Delta_1(t_0) - (j-1)p = 2p$, following from equation (4).

When $SUCC(j-1) = t_0 - 1$, $SUCC(j) = t_0$ and $SUCC(j+1) = t_0 + 1$, it follows from Proposition 1 that

$$(j-1)p < \Delta_1(t_0 - 1) \leq jp < \Delta_1(t_0) \leq (j+1)p. \tag{7}$$

Since both $\Delta_1(t_0 - 1)/p$ and $\Delta_1(t_0)/p$ are integers, along with equation (5), equation (7) implies $\Delta_1(j) - \Delta_1(j-1) = \Delta_1(t_0) - \Delta_1(t_0 - 1) = p$.

When $SUCC(j-1) = t_0 - 1$ and $SUCC(j) = SUCC(j+1) = t_0$, it follows from Proposition 1 that

$$(j-1)p < \Delta_1(t_0 - 1) \leq jp < (j+1)p < \Delta_1(t_0). \tag{8}$$

Since both $\Delta_1(t_0 - 1)/p$ and $\Delta_1(t_0)/p$ are integers, along with equation (3), equation (8) implies $\Delta_1(j) - \Delta_1(j-1) = (j+1)p - \Delta_1(t_0 - 1) = p$. □

Proposition 4 *Assume $S = vp$. Then*:

(i) $|N_k| = kv$ *for* $k = 1, 2, \ldots$.

(ii) $|C_{ki}| = k$ *for* $i = 1, \ldots, v$.
(iii) *For* $j \in N_k$, $\Delta_1(j) - \Delta_1(j-1) = 2p$ *if* j *is the last integer of* C_{ki} *for* $i = 1, \ldots, v$, $\Delta_1(j) - \Delta_1(j-1) = p$ *otherwise*.

Proof We prove this proposition by induction. Clearly, all these results hold when $k = 1$. Now suppose that they hold when $k = h$ for some positive integer h, *i.e.* $|N_h| = hv$, $|C_{hi}| = h$ for $i = 1, \ldots, v$ and for $j \in N_h$, $\Delta_1(j) - \Delta_1(j-1) = 2p$ if j is the last integer of C_{hi} for $i = 1, \ldots, v$; $\Delta_1(j) - \Delta_1(j-1) = p$ otherwise. We need to show that they hold when $k = h+1$. For any $i \in \{1, \ldots, r\}$, by the induction assumption, we assume $C_{hi} = (a+1, \ldots, a+h)$, where a is a given positive integer, and we have

$$\Delta_1(a+1) - \Delta_1(a) = \cdots = \Delta_1(a+h-1) - \Delta_1(a+h-2) = p$$

and

$$\Delta_1(a+h) - \Delta_1(a+h-1) = 2p.$$

Due to Proposition 1, $j \in C_{h+1,i}$ if and only if

$$\Delta_1(a) \leq jp < \Delta_1(a+h) = \Delta_1(a) + (h+1)p. \tag{9}$$

Noting the fact that $\Delta_1(a)/p$ is a positive integer, the number of positive integers satisfying equation (9) must be $h+1$. Thus, we have $|C_{h+1,i}| = h+1$ and $C_{h+1,i} = (b, b+1, \ldots, b+h)$, where $b = \Delta_1(a)/p$. Since $\Delta_1(a+1) - \Delta_1(a) = \cdots = \Delta_1(a+h-1) - \Delta_1(a+h-2) = p$, we have $SUCC(j) \neq SUCC(j-1)$ for $j = b, b+1, \ldots, b+h-1$. Since $\Delta_1(a+h) - \Delta_1(a+h-1) = 2p$, we have $SUCC(b+h) \neq SUCC(b+h-1)$. Thus,

$$\Delta_1(b) - \Delta_1(b-1) = \cdots = \Delta_1(b+h-1) - \Delta_1(b+h-2) = p$$

and

$$\Delta_1(b+h) - \Delta_1(b+h-1) = 2p,$$

which follows from Proposition 3. By the arbitrariness of i, result (i), result (ii) and result (iii) hold for $k = h+1$. □

We define $f(j) = (\Delta_1(j) - \Delta_1(j-1))/p$ and denote by $f(j)$ the node in Figure 1 instead of integer j in C_{ki}. Now we can give the triangle structure diagram in constant time in the case $S = vp$ (see Figure 2).

To use the information provided by the diagram, we denote by (k, i, w) the wth integer in the ith periodic set C_{ki} of the kth batch case set N_k, *i.e.* $(k, i, w) = \sum_{t=1}^{k-1} tv + (i-1)k + w$ under $S = vp$. For example, $(4, 3, 2) = \sum_{t=1}^{4-1} tv + (3-1)4 + 2 = 28$ if $v = 3$. Since $(C_{1,1}, \ldots, C_{1,v}, \ldots, C_{k,1}, \ldots, C_{k,v}, \ldots)$ is a partition of the set of position integers, we may write every position integer $n = \sum_{t=1}^{k-1} tv + (i-1)k + w$ as (k, i, w), where all k, i, and w with $k \geq 1$, $1 \leq i \leq v$ and $1 \leq w \leq k$ are integers. We define $(0, 0, 0) = 0$. Now we give one of our main results.

$$\begin{array}{ccccccccccccccc}
2 & & & & & & & \cdots & 2 & & & & & & \\
1 & 2 & & & & & & & 1 & 2 & & & & & \\
1 & 1 & 2 & & & & & & 1 & 1 & 2 & & & & \\
1 & 1 & 1 & 2 & & & & & 1 & 1 & 1 & 2 & & & \\
1 & 1 & 1 & 1 & 2 & & & & 1 & 1 & 1 & 1 & 2 & & \\
1 & 1 & 1 & 1 & 1 & 2 & & & 1 & 1 & 1 & 1 & 1 & 2 & \\
1 & 1 & 1 & 1 & 1 & 1 & 2 & \cdots & 1 & 1 & 1 & 1 & 1 & 1 & 2 \\
 & & & \cdots & & & & & & & & \cdots & & &
\end{array}$$

Figure 2 Triangle structure diagram in the case *S* = *vp*.

Theorem 2 *Assume $S = vp$. Then for an arbitrary positive integer (k, i, w), the following hold and are decided in constant time*:

(i)

$$SUCC(k,i,w) = \begin{cases} 0, & \textit{if } k = 1; \\ (k-1,i,w) & \textit{if } w \le k-1; \\ (k-1,i,w-1) & \textit{if } w = k, \end{cases}$$

(ii) *the optimal solution of (k, i, w)-jobs batch sizing problem*

$$b_t = \begin{cases} (k-t)v + i - 1 & \textit{for } t = 1, \dots, k-w; \\ (k-t)v + i & \textit{for } t = k-w+1, \dots, k, \end{cases} \tag{10}$$

where all k, i, and w with $k \ge 1$, $1 \le i \le v$, and $1 \le w \le k$ are integers.

Proof For any n, we can give another kind of expression (k, i, w) in constant time. Also, the optimal successor and b_t can be decided in constant time. Thus, the specific expression of the optimal solution is decided in constant time.

In the case $k = 1$, $(k, i, w) = (1, i, 1) = i < v + 1$. This implies $(k, i, w)p < (v + 1)p = \Delta_1(0)$. By Proposition 1, $SUCC(k, i, w) = 0$. In the case $k > 1$, result (i) clearly follows from Proposition 4 or Figure 2. Due to result 1,

$$\begin{aligned}
&SUCC(k-t+1,i,w) = (k-t,i,w) \quad \text{for } t = 1, \dots, k-w, \\
&SUCC(k-t+1,i,k-t+1) = (k-t,i,k-t) \quad \text{for } t = k-w+1, \dots, k-1, \\
&SUCC(1,i,1) = 0.
\end{aligned}$$

Thus,

$$\begin{aligned}
b_t &= (k-t+1,i,w) - (k-t,i,w) \\
&= \left(\sum_{h=1}^{k-t} hv + (i-1)(k-t+1) + w\right) - \left(\sum_{h=1}^{k-t-1} hv + (i-1)(k-t) + w\right) \\
&= (k-t)v + i - 1
\end{aligned}$$

for $t = 1, \ldots, k - w$,

$$\begin{aligned} b_t &= (k-t+1, i, k-t+1) - (k-t, i, k-t) \\ &= \left(\sum_{h=1}^{k-t} hv + (i-1)(k-t+1) + (k-t+1)\right) - \left(\sum_{h=1}^{k-t-1} hv + (i-1)(k-t) + (k-t)\right) \\ &= (k-t)v + i \end{aligned}$$

for $t = k - w + 1, \ldots, k - 1$,

$$b_k = (1, i, 1) - 0 = i. \qquad \square$$

For example, we consider an instance of the batch sizing problem with 100 jobs and $S = 4p$. 100 can be written as $84 + 14 + 2 = \sum_{t=1}^{7-1} t \times 4 + (3 - 1) \times 7 + 2 = (7, 3, 2)$. By the triangle structure diagram in the case $S = 4p$, we can obtain the successor relation between the number of jobs: $(7,3,2) \to (6,3,2) \to (5,3,2) \to (4,3,2) \to (3,3,2) \to (2,3,2) \to (1,3,1) \to (0,0,0)$. Thus, the optimal solution is $(b_1, \ldots, b_7) = (26, 22, 18, 14, 10, 7, 3)$. Of course, we may take the optimal solution by (10). For example, for a problem with 1,023 jobs and $S = 10p$, by $1{,}019 = 910 + 98 + 11 = \sum_{t=1}^{14-1} t \times 10 + (8 - 1) \times 14 + 11 = (14, 8, 11)$, the optimal solution $(b_1, \ldots, b_{14}) = (137, 127, 117, 108, 98, 88, 78, 68, 58, 48, 38, 28, 18, 8)$ as follows from (10).

We recall the solving formula by Santos [6] and Santos and Magazine [7]: $b_i = n/k + S(k+1)/2p - iS/p$ for $i = 1, \ldots, k$, where $k = \lceil \sqrt{1/4 + 2np/S} - 1/2 \rceil$. The validity of the formula depends on whether $n/k + S(k+1)/2p - iS/p$, for $i = 1, \ldots, k$, are integers. First, it is necessary that S/p is an integer, *i.e.* $S = vp$, which is the same as the case considered by us in this section. Noting the fact in the solving formula $b_1 - b_2 = \cdots = b_{k-1} - b_k = v$, it follows from the triangle structure diagram in the case $S = vt$ that $n = (k, i, 1)$ and/or $n = (k, i, k)$. If $n = (k, i, 1)$, then $n/k + S(k+1)/2p = (\sum_{t=1}^{k-1} tv + (i-1)k + 1)/k + (k+1)v/2 = (k-1)v/2 + i - 1 + 1/k + (k+1)v/2 = kv + i - 1 + 1/k$. Clearly, when $k > 1$, $n/k + S(k+1)/2p$ is not an integer otherwise. Case $k = 1$ can be reduced to the case $n = (k, i, k)$; if $n = (k, i, k)$, then $n/k + S(k+1)/2p = (\sum_{t=1}^{k-1} tv + (i-1)k + k)/k + (k+1)v/2 = (k-1)v/2 + i - 1 + 1 + (k+1)v/2 = kv + i$ is an integer. Thus, the solving formula by Santos [6] and Santos and Magazine [7] is appropriate for the problem with $S = vp$ and $n = \sum_{t=1}^{k-1} tv + (i-1)k + k$, where k, v, and $i \le v$ are positive integers.

4 The triangle structure diagram in the case $S = vp + r$

In this section we present a specific diagram of the batching problem with $S = vp + r$ that can be obtained in $O(\sqrt{n})$ time, where $0 < r < p$ and v are integers. Since the diagram consists of v the same triangles and has a triangle-like shape, we call it a triangle structure diagram. Using the optimal successor membership between integers presented by the diagram, we give the specific expression of the optimal solution for the any-number-jobs batching problem with $S = vp + r$.

Noting the fact that $\Delta_1(0) = S + p = (v+1)p + r$, we have $N_1 = (1, \ldots, v, v+1)$. We rewrite N_k as $N_k = (C_{k,1}, \ldots, C_{k,v}, A_k)$ for $k = 1, 2, \ldots$.

The diagram depends on the first-order difference sequence. Similar to the process in the last section, we first explore the properties of the first-order difference and then give the triangle structure diagram. The proposition below shows that the increments of the first-order difference for the first kv integers in N_k are of periodicity. Using this property, we can reduce the calculated amount of determining the first-order difference.

Proposition 5 *Assume* $S = vp + r$, *where* $0 < r < p$ *and* v *are integers, and let* $q_{k-1} = \sum_{t=1}^{k-1} |N_t|$. *Then*:

(i) $|C_{ki}| = k$ *for* $i = 1, \ldots, v$.

(ii) $\Delta_1(q_{k-1} + (i-1)k + k) - \Delta_1(q_{k-1} + (i-1)k) = (k+1)p$ *for* $i = 1, \ldots, v$.

(iii) $\Delta_1(q_{k-1} + ik + w) - \Delta_1(q_{k-1} + ik + (w-1)) =$
$\Delta_1(q_{k-1} + (i-1)k + w) - \Delta_1(q_{k-1} + (i-1)k + (w-1))$ *for* $i = 1, \ldots, v-1$ *and* $w = 1, \ldots, k$.

Proof We prove this proposition by induction. Clearly, all these results hold when $k = 1$. Now suppose that they hold when $k = h$ for some positive integer h. We need to show that they hold when $k = h + 1$. For any $i \in \{1, \ldots, v\}$, by result (ii) with $k = h$: $\Delta_1(q_{k-1} + (i-1)h + h) - \Delta_1(q_{k-1} + (i-1)h) = (h+1)p$, and it follows from Proposition 1 that the result (i) holds when $k = h + 1$.

Now we may write $C_{h+1,i}$ as $(b+1, \ldots, b+(h+1))$, where $b = q_h + (i-1)(h+1)$. By Proposition 2, every integer in $C_{h,i}$ must be the optimal successor of some integer in $C_{h+1,i}$ and there is an integer, say $a + t_0$, where $a = q_{h-1} + (i-1)h$, in $C_{h,i}$ such that it is the optimal successor of two integers in $C_{h+1,i}$. Thus, we have

$$SUCC(b+t) = a + t \quad \text{for } t = 1, \ldots, (t_0 - 1),$$

$$SUCC(b+t_0) = SUCC(b+t_0+1) = a + t_0,$$

$$SUCC(b+t) = a + t - 1 \quad \text{for } t = (t_0+2), \ldots, (h+1).$$

By equation (5),

$$\Delta_1(b+t) - \Delta_1(b+t-1) = \Delta_1(a+t) - \Delta_1(a+t-1) \tag{11}$$

for $t = 1, \ldots, (t_0 - 1)$.

By equation (3),

$$\Delta_1(b+t_0) - \Delta_1(b+t_0-1) = (b+t_0+1)p - \Delta_1(a+t_0-1). \tag{12}$$

By equation (4),

$$\Delta_1(b+t_0+1) - \Delta_1(b+t_0) = \Delta_1(a+t_0) - (b+t_0)p. \tag{13}$$

By equation (5),

$$\Delta_1(b+t) - \Delta_1(b+t-1) = \Delta_1(a+t-1) - \Delta_1(a+t-2) \tag{14}$$

for $t = (t_0+2), \ldots, (h+1)$. Equations (11), (12), (13), and (14), along with the induction assumption, imply $\Delta_1(b+h+1) - \Delta_1(b) = \Delta_1(a+h) - \Delta_1(a) + p = (h+2)p$, *i.e.* the result (ii) holds when $k = h + 1$.

By the induction assumption, we see that for, $i = 1, \ldots, v$, the position of the integer in each C_{hi} such that it is the optimal successor of two integers in $C_{h+1,i}$ is the same, respectively. Using a similar argument as in the proof of result (ii), we may obtain the respective formulas (11), (12), (13), and (14) for each $i \in \{1, \ldots, v\}$. The difference between them is that there are different a and b for different formulas. By comparing those formulas, along with the induction assumption, we see that result (iii) holds when $k = h + 1$. □

The proposition below shows the number of integers in N_k and the first-order difference of the last integer in N_k. We define $Q(j) = \Delta_1(j)/p$. In view of the fact that $\Delta_1(j)$ is an integer multiple of $Q(j)$ and the fact that all results above relative to the first-order difference hold if we replace 'Δ_1' by 'Q' and remove 'p', we call still $Q(j)$ as the first-order difference at j. For example, the sequence $\{Q(i)\}_{i=-1}^{+\infty}$ is strictly monotone increasing, equation (3) becomes $Q(h+1) - Q(h) = (h+2) - Q(t_0 - 1)$ and $\Delta_1(q_{k-1} + (i-1)k + k) - \Delta_1(q_{k-1} + (i-1)k) = (k+1)p$ implies $Q(q_{k-1} + (i-1)k + k) - Q(q_{k-1} + (i-1)k) = (k+1)$. We denote by $\lfloor x)$ the maximal integer less than the rational number x. Clearly, $0 < x - \lfloor x) \leq 1$, for example, $\lfloor 4.2) = 4$ and $\lfloor 4) = 3$.

Proposition 6 *Assume $S = vp + r$ and $\varepsilon = r/p$, where $0 < r < p$ and v are integers. Then for $k = 1, 2, \ldots$*

(i) $|A_k| = 1 + \lfloor k\varepsilon)$,
(ii) $q_k = k + \sum_{t=1}^{k}(tv + \lfloor t\varepsilon))$,
(iii) $Q(q_k) = \sum_{t=1}^{k}(tv + \lfloor t\varepsilon)) + (k+1) + (k+1)v + (k+1)\varepsilon$,

where $q_k = \sum_{t=1}^{k} |N_t|$ is the last integer in N_k.

Proof We prove this proposition by induction. $|A_1| = 1$ is known. Since $0 < \varepsilon < 1$, we have $\lfloor \varepsilon) = 0$. This implies that the results (i) and (ii) hold when $k = 1$. By Proposition 5, $Q(v) = Q(0) + 2v$. Since $SUCC(v) = SUCC(v+1) = 0$, $SUCC(v+2) = 1$ and $\lfloor \varepsilon) = 0$, by (4), we have $Q(q_1) = Q(v+1) = Q(v) + Q(0) - v = 2v + 2(v+1) + 2\varepsilon - v = v + 2 + 2v + 2\varepsilon$. This implies that result (iii) holds when $k = 1$. Now suppose that they hold when $k = h$ for some positive integer h. We need to show that they hold when $k = h + 1$. We know that $A_{h+1} = \{j \mid \Delta_1(q_h - |A_h|) \leq jp < \Delta_1(q_h)\}$ or $A_{h+1} = \{j \mid Q(q_h - |A_h|) \leq j < Q(q_h)\}$. By Proposition 5 and the induction assumption,

$$\begin{aligned} Q\bigl(q_h - |A_h|\bigr) &= Q(q_{h-1}) + (h+1)v = \sum_{t=1}^{h-1}\bigl(tv + \lfloor t\varepsilon)\bigr) + h + hv + h\varepsilon + (h+1)v \\ &= h + (h+1)v + \sum_{t=1}^{h}\bigl(tv + \lfloor t\varepsilon)\bigr) + \bigl(h\varepsilon - \lfloor h\varepsilon)\bigr) \end{aligned}$$

and

$$Q(q_h) = (h+1) + \sum_{t=1}^{h+1}\bigl(tv + \lfloor t\varepsilon)\bigr) + \bigl((h+1)\varepsilon - \lfloor (h+1)\varepsilon)\bigr).$$

By $0 < (h\varepsilon - \lfloor h\varepsilon)) \leq 1$ and $0 < ((h+1)\varepsilon - \lfloor (h+1)\varepsilon)) \leq 1$, we see that the minimal positive integer greater than or equal to $Q(q_h - |A_h|)$ is $h + (h+1)v + \sum_{t=1}^{h}(tv + \lfloor t\varepsilon)) + 1$, and the maximal positive integer less than $Q(q_h)$ is $(h+1)+\sum_{t=1}^{h+1}(tv+\lfloor t\varepsilon))$. Thus,

$$\begin{aligned}|A_{h+1}| &= (h+1)+\sum_{t=1}^{h+1}\bigl(tv+\lfloor t\varepsilon)\bigr)-\left(h+(h+1)v+\sum_{t=1}^{h}\bigl(tv+\lfloor t\varepsilon)\bigr)\right)\\ &= 1+\bigl\lfloor (h+1)\varepsilon\bigr).\end{aligned}$$

This implies result (i) holds. Due to Proposition 5, we have

$$\begin{aligned}q_{h+1} &= q_h+(h+1)v+1+(h+1)\bigl\lfloor (h+1)\varepsilon\bigr)\\ &= h+\sum_{t=1}^{h}\bigl(tv+\lfloor t\varepsilon)\bigr)+(h+1)v+1+\bigl\lfloor (h+1)\varepsilon\bigr)=(h+1)+\sum_{t=1}^{h+1}\bigl(tv+\lfloor t\varepsilon)\bigr).\end{aligned}$$

This implies result (i) holds. To prove result (iii) with $k=h+1$, we consider two cases: (a1) $\lfloor(h+1)\varepsilon)=\lfloor h\varepsilon)$, and (b1) $\lfloor(h+1)\varepsilon)=\lfloor h\varepsilon)+1$.

Case (a1) $\lfloor(h+1)\varepsilon)=\lfloor h\varepsilon)$. In this case, we have $|A_{h+1}|=|A_h|$. This implies that all the optimal successors of integers in A_{h+1} are different. By equation (5), $Q(q_{h+1})-Q(q_{h+1}-\lfloor(h+1)\varepsilon)-1)=Q(q_h)-Q(q_h-\lfloor(h)\varepsilon)-1)$. This, along with the induction assumption and Proposition 5, implies that

$$\begin{aligned}Q(q_{h+1}) &= Q\bigl(q_{h+1}-\bigl\lfloor (h+1)\varepsilon\bigr)-1\bigr)+Q(q_h)-Q\bigl(q_h-\lfloor h\varepsilon)-1\bigr)\\ &= Q(q_h)+(h+2)v+\bigl(Q(q_h)-Q(q_{h-1})\bigr)-(h+1)v\\ &= Q(q_h)+v+\lfloor h\varepsilon)+1+(h+2)v+\varepsilon\\ &= \sum_{t=1}^{h+1}\bigl(tv+\lfloor t\varepsilon)\bigr)+(h+2)+(h+2)v+(h+2)\varepsilon.\end{aligned}$$

Case (b1) $\lfloor(h+1)\varepsilon)=\lfloor h\varepsilon)+1$. In this case, we have $|A_{h+1}|=|A_h|+1$. By similar arguments as used in the proof of Proposition 5, $Q(q_{h+1})-Q(q_{h+1}-\lfloor(h+1)\varepsilon)-1)=Q(q_h)-Q(q_h-\lfloor(h)\varepsilon)-1)+1$. By the same argument as in Case (a1), we have

$$\begin{aligned}Q(q_{h+1}) &= 1+Q(q_h)+v+\lfloor h\varepsilon)+1+(h+2)v+\varepsilon\\ &= Q(q_h)+\bigl\lfloor (h+1)\varepsilon\bigr)+(h+1)v+1+v+\varepsilon\\ &= \sum_{t=1}^{h+1}\bigl(tv+\lfloor t\varepsilon)\bigr)+(h+2)+(h+2)v+(h+2)\varepsilon.\end{aligned}$$

This ends the proof for result (iii). □

The proposition below shows that the optimal successor membership between numbers of jobs is exactly determined by the coefficients of ε in the expression of the first-order difference. To determine the optimal successor membership between the numbers of jobs, by Proposition 5, we assume $v=1$. We denote by (k,w) the wth integer of C_{k1} and by $(k,k+w)$ the wth integer of A_k. We define $(k,0)=(k,1)-1$ and $(k,k+0)=(k,k)$, respectively. $c_{k,w}$ denotes the coefficient of ε in $Q(k,w)$ for $w=1,\dots,k+\lfloor k\varepsilon)+1$, or the coefficient of (k,w) for short. We set $r_k=\lfloor k\varepsilon)$.

Proposition 7 *Assume $S = vp + r$ and $\varepsilon = r/p$, where $0 < r < p$ and v are integers. Then for $k = 1, 2, \ldots$:*

(i) *$(c_{k1}, \ldots, c_{kk})$ is a permutation of $\{1, \ldots, k\}$.*

(ii) *If $c_{kt_0} = 1$, then $SUCC(k, t_0) = SUCC(k, t_0 + 1) = (k - 1, t_0)$; if $c_{k,k+t_0} = 1$, then $SUCC(k, t_0) = SUCC(k, t_0 + 1) = (k - 1, t_0)$.*

(iii) Triangle rule:

$$c_{k,w} = c_{k-1,r_{k-1}+w} \quad \text{for } w = 1, \ldots, k, \tag{15}$$

$$c_{k,k+w} = c_{kw} \quad \text{for } w = 1, \ldots, r_k, \tag{16}$$

$$c_{k,k+r_k+1} = k + 1. \tag{17}$$

Proof Since $F(0) = 0$, $F(1)/p = v + 1 + \varepsilon$ and $F(j)/p = j(v + \varepsilon + (j - SUCC(j))) + F(SUCC(j))/p$ for $j = 1, 2, \ldots$, $Q(j) = (F(j + 1) - F(j))/p$ is a linear function of ε. Now we prove this proposition by induction. Simple calculations yield

$$Q(0,0) = 2 + \varepsilon;$$
$$Q(1,1) = 4 + \varepsilon, \qquad Q(1,1+1) = 5 + 2\varepsilon;$$
$$Q(2,1) = 7 + \varepsilon, \qquad Q(2,2) = 8 + 2\varepsilon, \qquad Q(2,2+1) = 9 + 3\varepsilon$$

if $r_2 \leq 1$ and

$$Q(2,1) = 7 + \varepsilon, \qquad Q(2,2) = 8 + 2\varepsilon, \qquad Q(2,2+1) = 10 + \varepsilon, \qquad Q(2,2+2) = 10 + 3\varepsilon$$

if $r_2 > 1$. This implies that all results hold when $k = 1$ and $k = 2$. Now suppose that they hold when $k = h$ for some positive integer $h \geq 2$. We need to show that they hold when $k = h + 1$.

Now we prove result (i). There are two cases to consider: (a3) $SUCC(h+1,1) = SUCC(h+1,2) = (h,1)$, and (b3) $SUCC(h + 1, 1) \neq SUCC(h + 1, 2)$.

Case (a3) $SUCC(h + 1, 1) = SUCC(h + 1, 2) = (h, 1)$. By equation (3), $Q(h + 1, 1) = Q(h, h + (1 + r_h)) + (h + 1, 2) - Q(h - 1, (h - 1) + (1 + r_{h-1}))$. Due to the induction assumption, $c_{h,h+(1+r_h)} = h + 1$ and $c_{h-1,(h-1)+(1+r_{h-1})} = h$. Thus, we have $c_{h+1,1} = 1$.

By equation (4), $Q(h + 1, 2) = Q(h + 1, 1) + Q(h, 1) - (h + 1, 1)$. Thus, we have $c_{h+1,2} = c_{h+1,1} + c_{h,1} = c_{h,1} + 1$.

By equation (5), $Q(h + 1, 3) = Q(h + 1, 2) + Q(h, 2) - Q(h, 1)$. Thus, we have $c_{h+1,3} = c_{h+1,2} + c_{h,2} - c_{h,1} = c_{h,1} + 1 + c_{h,2} - c_{h,1} = c_{h,2} + 1$. Similarly, we have $c_{h+1,w} = c_{h,w-1} + 1$ for $w = 4, \ldots, h+1$. Thus,

$$(c_{h+1,1}, \ldots, c_{h+1,h+1}) = (1, c_{h,1} + 1, c_{h,2} + 1, \ldots, c_{h,h} + 1). \tag{18}$$

This, along with the induction assumption, implies that $(c_{h+1,1}, \ldots, c_{h+1,h+1})$ is a permutation of $\{1, \ldots, h + 1\}$. Thus, result (i) holds in the case.

Case (b3) $SUCC(h + 1, 1) \neq SUCC(h + 1, 2)$. We assume $SUCC(h + 1, t_0) = SUCC(h + 1, t_0 + 1) = (h, t_0)$, where $1 < t_0 \leq h$. By equation (5), $Q(h + 1, 1) = Q(h, h + (1 + r_h)) + Q(h, 1) - Q(h - 1, (h - 1) + (1 + r_{h-1}))$. Due to the induction assumption, $c_{h+1,1} = c_{h,h+(1+r_h)} +$

$c_{h,1} - c_{h-1,(h-1)+(1+r_{h-1})} = (h+1) + c_{h,1} - h = c_{h,1} + 1$. Similarly, we have $c_{h+1,w} = c_{h,w} + 1$ for $w = 2, \ldots, t_0 - 1$.

By equation (3), $Q(h+1, t_0) = Q(h+1, t_0 - 1) + (h+1, t_0 + 1) - Q(h, t_0 - 1)$. Thus, we have $c_{h+1,t_0} = c_{h+1,t_0-1} - c_{h,t_0-1} = 1$.

By equation (4), $Q(h+1, t_0+1) = Q(h+1, t_0) + Q(h, t_0) - (h+1, t_0 - 1)$. Thus, we have $c_{h+1,t_0+1} = c_{h+1,t_0} + c_{h,t_0} = c_{h,t_0} + 1$. Similarly, we have $c_{h+1,w} = c_{h,w-1} + 1$ for $w = t_0 + 2, \ldots, h+1$. Thus,

$$(c_{h+1,1}, \ldots, c_{h+1,h+1}) = (c_{h,1} + 1, \ldots, c_{h,t_0-1} + 1, 1, c_{h,t_0} + 1, \ldots, c_{h,h} + 1). \tag{19}$$

This, along with the induction assumption, implies that $(c_{h+1,1}, \ldots, c_{h+1,h+1})$ is a permutation of $\{1, \ldots, h+1\}$. Thus, result (i) holds in the case.

By the same arguments as in the proof of result (i), we have

$$(c_{h+1,(h+1)+1}, \ldots, c_{h+1,(h+1)+(r_{h+1}+1)}) = (c_{h,h+1} + 1, \ldots, c_{h,h+(r_h+1)} + 1) \tag{20}$$

if $r_{h+1} = r_h$;

$$\begin{aligned}
&(c_{h+1,(h+1)+1}, \ldots, c_{h+1,(h+1)+(r_{h+1}+1)}) \\
&\quad = (c_{h,h+1} + 1, \ldots, c_{h,h+(t_0-1)} + 1, 1, c_{h,h+(t_0)} + 1, \ldots, c_{h,h+(r_h+1)} + 1)
\end{aligned} \tag{21}$$

if $r_{h+1} = r_h + 1$, where $1 \le t_0 \le (r_h + 1)$. Due to equations (18), (19), (20), and (21), result (ii) holds when $k = h+1$.

Before other results are proved, we give an assertion, as follows.

Assume that c_{j-1} and c_j are the coefficients of ε corresponding to $Q(j-1)$ and $Q(j)$ respectively, where $j \in N_k$ and $k \ge 2$. Then

($*$) j is the optimal successor of two integers if $c_{j-1}\varepsilon - \lfloor c_{j-1}\varepsilon) > c_j\varepsilon - \lfloor c_j\varepsilon)$;
($**$) j is the optimal successor of one integer if $c_{j-1}\varepsilon - \lfloor c_{j-1}\varepsilon) \le c_j\varepsilon - \lfloor c_j\varepsilon)$.

We prove this assertion by contradiction. We write $Q(j-1)$ and $Q(j)$ as $a + c_{j-1}\varepsilon$ and $b + c_j\varepsilon$, respectively, where a and b are integers. Clearly, $a + \lfloor c_{j-1}\varepsilon) + 1$ is the minimal integer greater than or equal to $Q(j-1)$ and $b + \lfloor c_j\varepsilon)$ the maximal positive integer less than $Q(j)$. Suppose that result ($*$) does not hold. Then we have $b + \lfloor c_j\varepsilon) - (a + \lfloor c_{j-1}\varepsilon)) = 1$. By $c_{j-1}\varepsilon - \lfloor c_{j-1}\varepsilon) > c_j\varepsilon - \lfloor c_j\varepsilon)$, $Q(j) = b + \lfloor c_j\varepsilon) + (c_j\varepsilon - \lfloor c_j\varepsilon)) < a + \lfloor c_{j-1}\varepsilon) + (c_{j-1}\varepsilon - \lfloor c_{j-1}\varepsilon)) + 1 = Q(j-1) + 1$, *i.e.* $Q(j) - Q(j-1) < 1$. This contradicts Proposition 2.

Suppose that result ($**$) does not hold. Then $b + \lfloor c_j\varepsilon) - (a + \lfloor c_{j-1}\varepsilon)) = 2$. If $c_{j-1}\varepsilon - \lfloor c_{j-1}\varepsilon) < c_j\varepsilon - \lfloor c_j\varepsilon)$, using a similar argument as in the proof of result ($*$), we can deduce that $Q(j) - Q(j-1) > 2$. This contradicts Proposition 2. If $c_{j-1}\varepsilon - \lfloor c_{j-1}\varepsilon) = c_j\varepsilon - \lfloor c_j\varepsilon)$, then $Q(j) - Q(j-1) = 2$. In the case $j \in C_{k1}$, this, along with Proposition 5, implies $Q(i) - Q(i-1) = 1$ for an arbitrary $i \in C_{k1}$ with $i \neq j$. And then we have $c_{i-1}\varepsilon - \lfloor c_{i-1}\varepsilon) = c_i\varepsilon - \lfloor c_i\varepsilon)$. Since $\{c_j \mid j \in C_{k1}\} = \{1, 2, \ldots, k\}$, we have $\varepsilon - \lfloor \varepsilon) = 2\varepsilon - \lfloor 2\varepsilon)$. This contradicts $0 < \varepsilon < 1$. In the case $j \in A_k$, $Q(j) - Q(j-1) = 2$ contradicts the fact that $Q(q_k) - Q(q_k - r_k - 1) = r_k + 1 + \varepsilon$ and $Q(i) - Q(i-1) \ge 1$ for any integer i.

This assertion, along with the arguments used in the proof of result (i), implies that $(c_{h+1,1}, \ldots, c_{h+1,(h+1)+r_{h+1}+1})$ is exactly determined by $(c_{h,1}, \ldots, c_{h,h+r_h+1})$ and $c_{h-1,(h-1)+(r_{h-1}+1)}$.

We proceed with our proof. Note the fact that the coefficient permutations of the optimal successors corresponding to $\{(h, r_h + 1), \ldots, (h, h + r_h + 1)\}$ and $\{(h + 1, 1), \ldots, (h + 1, h + 1)\}$ are $(c_{h-1,r_{h-1}+1}, \ldots, c_{h-1,(h-1)+(r_{h-1}+1)})$ and $(c_{h,1}, \ldots, c_{h,h})$, respectively. By the induction assumption, we have $(c_{h-1,r_{h-1}+1}, \ldots, c_{h-1,(h-1)+(r_{h-1}+1)}) = (c_{h,1}, \ldots, c_{h,h})$ and both are the same permutation of $\{1, \ldots, h\}$. This, along with the assertion, implies that either

(a4) $c_{h+1,w} = c_{h,r_h+w} = c_{h,w-1} + 1$ for $w = 3, \ldots, h + 1$

or

(b4) $c_{h+1,w} = c_{h,r_h+w} = c_{h,w} + 1$ for $w = 2, \ldots, t_0 - 1$, $c_{h+1,t_0} = c_{h,r_h+t_0} = 1$ and $c_{h+1,w} = c_{h,r_h+w} = c_{h,w-1} + 1$ for $w = t_0 + 1, \ldots, h + 1$, where $2 \le t_0 \le h$ and $c_{h,t_0-1}\varepsilon - \lfloor c_{h,t_0-1}\varepsilon \rfloor > c_{h,t_0}\varepsilon - \lfloor c_{h,t_0}\varepsilon \rfloor$.

Since both $(c_{h+1,1}, \ldots, c_{h+1,h+1})$ and $(c_{h,r_h+1}, \ldots, c_{h,h+r_h+1})$ are permutations of $\{1, \ldots, h + 1\}$, along with result (ii) and the fact that $(c_{h,1}, \ldots, c_{h,h})$ is a permutation of $\{1, \ldots, h\}$, we have $c_{h+1,1} = c_{h,r_h+1} = 1$, $c_{h+1,2} = c_{h,r_h+2} = c_{h,1} + 1$ in the case (a4) and $c_{h+1,1} = c_{h,r_h+1} = c_{h,1} + 1$ in the case (b4). Thus, equation (15) holds when $k = h + 1$.

Now we prove equation (16). There are two cases to consider: (a5) $r_{h+1} = r_h$ and (b5) $r_{h+1} = r_h + 1$.

Case (a5) $r_{h+1} = r_h$. By equation (20), $c_{h+1,(h+1)+w} = c_{h,h+w} + 1$ for $w = 1, \ldots, r_{h+1}$. By the induction assumption, $c_{h,h+w} = c_{h,w}$ for $w = 1, \ldots, r_h$. This, along with $c_{h,h} = c_{h-1,(h-1)+r_{h-1}+1} = k$ and the assertion, implies that $c_{h+1,w} = c_{h,h+w} + 1 = c_{h+1,(h+1)+w}$ for $w = 1, \ldots, r_{h+1}$. Thus, equation (16) holds in the case.

Case (b5) $r_{h+1} = r_h + 1$. By equation (21), $c_{h+1,(h+1)+w} = c_{h,h+w} + 1$ for $w = 1, \ldots, t_0 - 1$, $c_{h+1,(h+1)+t_0} = 1$ and $c_{h+1,(h+1)+w} = c_{h,h+w-1} + 1$ for $w = t_0 + 1, \ldots, r_{h+1}$, where $1 \le t_0 \le (r_h + 1)$ If $1 \le t_0 \le r_h$, using similar arguments as in Case (a5), we find that equation (16) holds in the case. If $t_0 = r_h + 1$, then $c_{h+1,(h+1)+r_{h+1}} = 1$. This, along with equation (15), implies that $(c_{h+1,r_{h+1}+1}, \ldots, c_{h+1,(h+1)+r_{h+1}+1})$ is a permutation of $\{1, \ldots, h + 2\}$ and $1 \notin \{c_{h+1,r_{h+1}+1}, \ldots, c_{h+1,h+1}\}$. Using similar arguments as in Case (a5), we find that $1 < c_{h+1,w} = c_{h+1,(h+1)+w} = c_{h,w} + 1$ for $w = 1, \ldots, r_{h+1} - 1$. It follows from $(c_{h+1,1}, \ldots, c_{h+1,h+1})$ is a permutation of $\{1, \ldots, h + 1\}$ that $c_{h+1,r_{h+1}} = 1$. Thus, equation (16) holds in the case.

Equation (17) has been proven in Proposition 6. □

We denote by the coefficient of ε corresponding to integer j the node in Figure 1 instead of the integer j in C_{ki}. Using the triangle rule in Proposition 7, we may give a triangle structure diagram in the case $S = vp + r$.

We now show that the triangle structure diagram can be decided in $O(\sqrt{n})$ time. By the triangle rule in Proposition 7, we see that given the permutation of the numbers in the front row, the permutation of the numbers in the back row can be decided in constant time. The total of the permutations of k rows needs to be decided and the permutation of the first row is known. Because the number of batches in an optimal solution turns out to be $O(\sqrt{n})$, the triangle structure diagram can be decided in $O(\sqrt{n})$ time.

In order to better understand the characters of the triangle structure diagram, we give an example. Given $S = 241$ and $p = 100$, then $\varepsilon = 0.41$, and the triangle structure diagram of the batching problem is given in Figure 3

From the arguments above, we see clearly that the optimal successor membership between the numbers of jobs can exactly be determined by the position of '1' in $(c_{k,1}, \ldots, c_{k,k})$ and $(c_{k,k+1}, \ldots, c_{k,k+r_k+1})$. We denote by $c(k)$ and $a(k)$ the positions in $(c_{k,1}, \ldots, c_{k,k})$ and $(c_{k,k+1}, \ldots, c_{k,k+r_k+1})$ where 1 lies, respectively. We define $a(k) = +\infty$ if there is not 1 in

1										1										2				
1	2									1	2									3				
1	2	3								1	2	3								1	4			
2	3	1	4							2	3	1	4							2	5			
3	1	4	2	5						3	1	4	2	5						3	1	6		
4	2	5	3	1	6					4	2	5	3	1	6					4	2	7		
5	3	1	6	4	2	7				5	3	1	6	4	2	7				5	3	8		
1	6	4	2	7	5	3	8			1	6	4	2	7	5	3	8			1	6	4	9	
2	7	5	3	8	1	6	4	9		2	7	5	3	8	1	6	4	9		2	7	5	10	
3	8	1	6	4	9	2	7	5	10	3	8	1	6	4	9	2	7	5	10	3	8	1	6	11
			..	..	..								..	..	..						..	..	..	

Figure 3 Triangle structure diagram in the case $S = vp + r = 2 \times 100 + 41$.

$(c_{k,k+1}, \ldots, c_{k,k+r_k+1})$, and $r_0 = 0$. Using the triangle rule, a recursion for determining the position of 1 may be formulated as follows:

$$c(1) = 1;$$

$$c(k) = \begin{cases} c(k-1) - r_{k-1} & \text{if } r_{k-1} = r_{k-2}; \\ (k-1) + c(k-1) - r_{k-1} & \text{if } r_{k-1} = r_{k-2} + 1 \end{cases} \tag{22}$$

for $k = 2, 3, \ldots$,

$$a(k) = \begin{cases} +\infty & \text{if } r_k = r_{k-1}; \\ c(k) & \text{if } r_k = r_{k-1} + 1 \end{cases} \tag{23}$$

for $k = 1, 2, \ldots$.

Now we show that the recursion is correct. When $r_{k-1} = r_{k-2}$, by equation (16), $c(k-1) > r_{k-1}$. By equation (15), $c(k) = c(k-1) - r_{k-1}$. When $r_{k-1} = r_{k-2} + 1$, by equation (16), $a(k-1) = c(k-1)$. By equation (16), $c(k) = k - ((r_{k-1} + 1) - c(k-1)) = (k-1) + c(k-1) - r_{k-1}$. Thus, equation (22) holds for $k = 2, 3, \ldots$. Equation (23) follows from equation (16) for $k = 1, 2, \ldots$.

By the recursion above, we see that the positions of '1' in each row can be decided in constant time. The total of the positions of '1' in k rows needs to be decided and the position of '1' in the first row is known. Thus, all of the positions of '1' in the triangle structure diagram can be decided in $O(\sqrt{n})$ time.

We denote by (k, i, w_k) and (k, w_k) the w_kth integer in the ith periodic set C_{ki} of the kth batch case set N_k, and the w_kth integer in A_k, respectively, *i.e.* $(k, i, w_k) = \sum_{t=1}^{k-1} |N_t| + (i-1)k + w_k$ and $(k, w_k) = \sum_{t=1}^{k-1} |N_t| + vk + w_k$. Now we give the last result as follows.

Theorem 3 *Assume $S = vp + r$ with $0 < r < p$. Then for arbitrary positive integers (k, i, w_k) and (k, w_k), the following hold and are decided in $O(\sqrt{n})$ time*:

(i)

$$SUCC(k, i, w_k) = \begin{cases} 0 & \textit{if } k = 1; \\ (k-1, i, w_k) & \textit{if } w_k \le c(k); \\ (k-1, i, w_k - 1) & \textit{if } w_k > c(k). \end{cases}$$

(ii)

$$SUCC(k, w_k) = \begin{cases} 0 & \textit{if } k = 1; \\ (k-1, w_k) & \textit{if } w_k \le a(k); \\ (k-1, w_k - 1) & \textit{if } w_k > a(k). \end{cases}$$

(iii) *The optimal solution of the* (k, i, w_k)*-jobs batch sizing problem is*

$$b_j = \begin{cases} (k-j)v + r_{k-j} + i & \textit{if } w_{k+1-j} \le c(k+1-j); \\ (k-j)v + r_{k-j} + i + 1 & \textit{if } w_{k+1-j} > c(k+1-j) \end{cases} \tag{24}$$

for $j = 1, \dots, k$, *where* $c(j)$ *is given by equation* (22), *and* $w_{j-1} = w_j$ *if* $w_j \le c(j)$ *and* $w_{j-1} = w_j - 1$ *if* $w_j > c(j)$ *for* $j = k, \dots, 2$.

(iv) *The optimal solution of* (k, w_k)*-jobs batch sizing problem is*

$$b_j = \begin{cases} (k+1-j)v + r_{k-j} + 1 & \textit{if } w_{k+1-j} \le a(k+1-j); \\ (k+1-j)v + r_{k-j} + 2 & \textit{if } w_{k+1-j} > a(k+1-j) \end{cases} \tag{25}$$

for $j = 1, \dots, k$, *where* $a(j)$ *is given by equation* (23), *and* $w_{j-1} = w_j$ *if* $w_j \le a(j)$ *and* $w_{j-1} = w_j - 1$ *if* $w_j > a(j)$ *for* $j = k, \dots, 2$.

Proof For any $j = 1, \dots, k$, given $c(j)$ and $a(j)$, we can decide its the optimal successor and b_j in constant time. Since $c(j)$ and $a(j)$ can be decided in $O(\sqrt{n})$ time, the specific expression of the optimal solution is decided in $O(\sqrt{n})$ time.

In the case $k = 1$, $(k, i, w_k) = (1, i, 1) = i < v + 1 + \varepsilon = Q(0)$. By Proposition 1, $SUCC(k, i, w_k) = 0$. In the case $k > 1$, result (i) clearly follows from Proposition 7 or Figure 3. Similarly, result (ii) holds.

Now we show result (iii) for the case with $j = 1$. When $w_k \le c(k)$, due to result (i), $SUCC(k, i, w_k) = (k-1, i, w_k)$. Thus,

$$\begin{aligned} b_1 &= (k, i, w_k) - (k-1, i, w_k) \\ &= \left(\sum_{h=1}^{k-1} |N_h| + (i-1)k + w_k \right) - \left(\sum_{h=1}^{k-2} |N_h| + (i-1)(k-1) + w_k \right) \\ &= |N_{k-1}| + (i-1) \\ &= (k-1)v + r_{k-1} + i. \end{aligned}$$

When $w_k > c(k)$, due to result (i), $SUCC(k, i, w_k) = (k-1, i, w_k - 1)$. Thus,

$$\begin{aligned} b_1 &= (k, i, w_k) - (k-1, i, w_k - 1) \\ &= \left(\sum_{h=1}^{k-1} |N_h| + (i-1)k + w_k \right) - \left(\sum_{h=1}^{k-2} |N_h| + (i-1)(k-1) + (w_k - 1) \right) \\ &= |N_{k-1}| + (i-1) + 1 \\ &= (k-1)v + r_{k-1} + i + 1. \end{aligned}$$

The result for $j = 2, \dots, k-1$ can be proved similarly. The result for $t = k$ is

$$b_k = (1, i, 1) - 0 = i.$$

This finishes the proof of result (iii).

Using similar arguments as in the proof of result (iii), result (iv) can be proved. □

For example, we consider the instance of a batch sizing problem with 105 jobs, $S = 241$ and $p = 100$. Since 105 can be written as $105 = 91 + 9 + 5 = \sum_{h=1}^{9-1} |N_h| + (2-1)9 + 5 = (9, 2, 5)$. By the triangle structure diagram in the case $S = vp + r = 2 \times 100 + 41$, we can obtain the successor relation between the number of jobs: $8 \to 7 \to 6 \to 6 \to 5 \to 4 \to 3 \to 2 \to 1 \to 1$, *i.e.*, $(9,2,5) \to (8,2,5) \to (7,2,4) \to (6,2,3) \to (5,2,3) \to (4,2,2) \to (3,2,2) \to (2,2,1) \to (1,2,1) \to (0,0,0)$. Thus, the optimal solution is $(b_1, \dots, b_9) = (21, 19, 17, 14, 12, 9, 7, 4, 2)$. Of course, we may have the optimal solution by equation (24). However, before we solve the instance, we need to go to computer r_j, $c(j)$, $a(j)$ and $\sum_{h=1}^{j} |N_h|$ as follows:

level j:	0	1	2	3	4	5	6	7	8	9	10	$\cdots$;
r_j	0	0	0	1	1	2	2	2	3	3	4	$\cdots$;
$c(j)$		1	1	1	3	2	5	3	1	6	3	$\cdots$;
$a(j)$		$+\infty$	$+\infty$	1	$+\infty$	2	$+\infty$	$+\infty$	1	$+\infty$	3	$\cdots$;
$\sum_{h=1}^{j} \lvert N_h \rvert$		3	8	16	26	39	54	71	91	113	138	$\cdots$.

Using equation (24), we have

by $w_9 = 5 \le c(9) = 6$, $b_1 = (9-1)2 + 3 + 2 = 21$ and $w_8 = w_9 = 5$;
by $w_8 = 5 > c(8) = 1$, $b_2 = (9-2)2 + 2 + 2 + 1 = 19$ and $w_7 = w_8 - 1 = 4$;
by $w_7 = 4 > c(7) = 3$, $b_3 = (9-3)2 + 2 + 2 + 1 = 17$ and $w_6 = w_7 - 1 = 3$;
by $w_6 = 3 \le c(6) = 5$, $b_4 = (9-4)2 + 2 + 2 = 14$ and $w_5 = w_6 = 3$;
by $w_5 = 3 > c(5) = 2$, $b_5 = (9-5)2 + 1 + 2 + 1 = 12$ and $w_4 = w_5 - 1 = 2$;
by $w_4 = 2 \le c(4) = 3$, $b_6 = (9-6)2 + 1 + 2 = 9$ and $w_3 = w_4 = 2$;
by $w_3 = 2 > c(3) = 1$, $b_7 = (9-7)2 + 0 + 2 + 1 = 7$ and $w_2 = w_3 - 1 = 1$;
by $w_2 = 1 \le c(2) = 1$, $b_8 = (9-8)2 + 0 + 2 = 4$;
$b_9 = 2$.

If n is chosen $137 = 113 + 20 + 4 = \sum_{h=1}^{10-1} |N_h| + 2 \times 10 + 4 = (10, 4)$, then using equation (25), similarly we have $(b_1, \dots, b_{10}) = (25, 22, 19, 17, 15, 13, 10, 8, 5, 3)$.

5 Conclusions

In this paper the problem of batching identical jobs on a single machine to minimize the completion time is studied by employing the difference analysis technique. We first establish the relation between the optimal solution and the first-order difference of the optimal objective function in terms of the number of jobs and investigate the properties of the first-order difference. Then, we obtain the triangle structure diagram of the batching problem in $O(\sqrt{n})$ time at most by using the permutation of some numbers which describe the character of the first-order difference. The diagrams enable us to see clearly the specific expressions of optimal solutions for the n-jobs batching problem and any m-jobs batching problem simultaneously, where $m \le n$. The difference analysis technique employed by us should be used for solving other batching problems. Analysis of many batching problems shows that, in some cases, the sequencing and batching of jobs can be decoupled. Once a sequence of jobs is known, dynamic programming is often adopted to solve the batching problem. In the case the difference expression of the optimal solution function may be obtained. It become possible that the solving algorithm may be improved by employing the difference analysis technique. The difference analysis technique should be a better tool for solving discrete mathematics problems.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
All authors contributed equally to the manuscript and typed, read, and approved the final manuscript.

Author details
[1]Department of Mathematics, Taizhou University, Taizhou, Zhejiang 317000, P.R. China. [2]Department of Logistics and Maritime Studies, Hong Kong Polytechnic University, Kowloom, Hong Kong, P.R. China.

Acknowledgements
This research was supported in part by the Zhejiang Natural Science Foundation of China Grant Y6110054, the Shanxi Scholarship Council of China, and the Hong Kong Polytechnic University Research Grant J-BB7D and The Hong Kong CERG Research Fund 5515/10H.

Received: 11 July 2013 Accepted: 26 December 2013 Published: 19 February 2014

References
1. Coffman, EG Jr., Yannakakis, M, Magazine, MJ, Santos, CA: Batch sizing and job sequencing on a single machine. Ann. Oper. Res. **26**, 135-147 (1990)
2. Naddef, D, Santos, C: One-pass batching algorithms for the one-machine problem. Discrete Appl. Math. **21**, 133-145 (1988)
3. Coffman, EG Jr., Nozari, A, Yannakakis, M: Optimal scheduling of products with two subassemblies on a single machine. Oper. Res. **37**, 426-436 (1989)
4. Shallcross, D: A polynomial algorithm for a one machine batching problem. Oper. Res. Lett. **11**, 213-218 (1992)
5. Potts, CN, Kovalyov, MY: Scheduling with batching: a review. Eur. J. Oper. Res. **120**, 228-249 (2000)
6. Santos, CA: Batching and sequencing decisions under lead time considerations for single machine problems. MSc thesis, Department of Management Sciences, University of Waterloo, Canada (1984)
7. Santos, CA, Magazine, M: Batching in single operation manufacturing systems. Oper. Res. Lett. **4**, 99-103 (1985)
8. Albers, S, Brucker, P: The complexity of one-machine batching problems. Discrete Appl. Math. **47**, 87-107 (1993)

doi:10.1186/1687-1847-2014-71
Cite this article as: Li et al.: Triangle structure diagrams for a single machine batching problem with identical jobs. *Advances in Difference Equations* 2014 **2014**:71.