Available online at www.sciencedirect.com
ScienceDirect
Discrete Applied Mathematics 155 (2007) 2140–2151
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

On packing and coloring hyperedges in a cycle

Jianping Li[a,*], Lusheng Wang[b], Hao Zhao[b]

[a]Department of Mathematics, Yunnan University, Kunming 650091, PR China
[b]Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong, PR China

Received 10 June 2004; received in revised form 9 May 2007; accepted 15 May 2007
Available online 8 June 2007

Abstract

Given a hypergraph and k different colors, we study the problem of packing and coloring a subset of the hyperedges of the hypergraph as paths in a cycle such that the total profit of the hyperedges selected is maximized, where each physical link e_j on the cycle is used at most c_j times, each hyperedge h_i has its profit p_i and any two paths, each spanning all nodes of its corresponding hyperedge, must be assigned different colors if they share a common physical link. This new problem arises in optical communication networks, and it is called the *Maximizing Profits when Packing and Coloring Hyperedges in a Cycle* problem (MPPCHC).

In this paper, we prove that the MPPCHC problem is NP-hard and then present an algorithm with approximation ratio 2 for this problem. For the special case where each hyperedge has the same profit 1 and each link e_j has same capacity k, we propose an algorithm with approximation ratio $\frac{3}{2}$.

MSC: 90B10; 94C15

Keywords: Hyperedge; Path coloring; Approximation algorithm

1. Introduction

Ganley and Cohoon [7] proposed the *Minimum-Congestion Hypergraph Embedding in a Cycle* problem (MCHEC), the objective is to embed *all* hyperedges of a hypergraph as paths in a cycle such that the congestion, i.e., the maximum number of paths over any physical link of the cycle, is minimized, where the hypergraph has the same nodes as the cycle and a path spans all nodes of its corresponding hyperedge. The MCHEC problem is a challenging one with applications in various areas such as computer networks, multicast communication, parallel computation, electronic design automation.

For general hypergraphs, Ganley and Cohoon [7] proved that the MCHEC problem is NP-hard. When a cycle consists of n nodes, and the m hyperedges in the hypergraph are defined on the nodes in the cycle, they provided a 3-approximation algorithm for the problem and gave an algorithm in time $\mathcal{O}((nm)^{k+1})$ to determine whether or not the problem has an embedding with congestion k. In this paper, we refer to an algorithm with approximation ratio k as a k-approximation algorithm. Gonzalez [8] derived two 2-approximation algorithms for this problem. One method transforms it to a linear integer programming problem and then uses an LP-based rounding technique. The other method transforms it to the

* Corresponding author. Tel.: +86 871 5032704; fax: +86 871 5033701.
E-mail addresses: jianping@ynu.edu.cn (J. Li), cswangl@cityu.edu.hk (L. Wang), zhaoh@cs.cityu.edu.hk (H. Zhao).

0166-218X/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2007.05.037

problem, where the hyperedges have exactly two nodes, which can be solved in linear time by the algorithm [6]. By utilizing an LP-based rounding algorithm, Lee and Ho [13] developed a linear time approximation algorithm to provide an embedding with congestion at most two times the optimum for the weighted version of the problem. Gu and Wang [9] gave a 1.8-approximation algorithm for the problem, with time complexity $\mathcal{O}(mn)$ which is best possible for hypergraphs with m hyperedges and n nodes. Lee and Ho [10,14] also designed a 1.5-approximation algorithm for the unweighted version of the problem, by using an LP-based algorithm, and a $(1.5+\varepsilon)$-approximation algorithm for the weighted version of the problem. Deng and Li [4] proposed a polynomial-time approximation scheme (PTAS) for the MCHEC problem based on a randomized rounding approach.

For the version where each hyperedge exactly consists of two nodes, the MCHEC problem becomes the ring loading problem [6,11,17]. Frank et al. [6] designed an optimal polynomial-time algorithm for the unweighted version of the ring load problem; however, Schrijver et al. [17] proved that the weighted version of the ring load problem is *NP*-hard and they developed an efficient algorithm to generate a solution that exceeds the optimum by at most an additive term of 1.5 times the maximum demand, and Khanna [11] proposed a PTAS that computes a solution with density at most $(1+\varepsilon)$ times the optimum one for any $\varepsilon > 0$.

In the application of the MCHEC problem, where the objective is to set up a connection to all nodes of a hyperedge, a path is selected from all possible c-paths, each c-path spanning all nodes of its corresponding hyperedge, and a wavelength is assigned to every physical link in the path. In all cases, if two paths share a common physical link, they must be assigned different wavelengths. This requirement is known as the *wavelength-continuity* constraint.

Similar to the ring loading problem, by considering the addition of slots (equivalently wavelengths) constraints, Carpenter et al. [2] proposed the demand routing and slotting problem (DRSP) on rings: given a set of requests (i.e., each request consists of two nodes) on the ring, what is the minimum number of slots (wavelengths) needed so that (i) each request is routed, (ii) each route assigned a slot, and (iii) no two routes that use the same physical link of the ring are assigned the same slot? They presented several 2-approximation algorithms for the DRSP problem. Kumar [12] utilized a randomization method to design a Monte Carlo approximation algorithm with approximation ratio $1.5 + 1/2e + o(1)$ for the DRSP problem, where e is the base of the natural logarithm. This randomized algorithm is a 1.68-randomized approximation algorithm. Cheng [3] presented the first deterministic approximation algorithm for the DRSP problem with an approximation ratio less than 2. Let w be the size of a largest set of request chords that mutually intersect at a point in the interior of the ring, Cheng showed that it is always possible to route and slot all requests using at most $(2 - 1/\lceil w/2 \rceil) \times OPT$ slots in an n-node network in time $\mathcal{O}(|I|n^2)$, where I is the set of all requests. Since request chords are incident to two nodes of the ring, then $w \leqslant \lfloor n/2 \rfloor$ holds, hence Cheng's algorithm achieves an approximation ratio of $2 - 8/(2n+4)$. In addition, when OPT is not large enough, the o(1) additive term in the Kumar's algorithm becomes significant so that the approximation ratio exceeds 2, but Cheng designed a randomized algorithm with constant approximation ratio $2 - 1/\theta(\log n)$, by using Kumar's algorithm.

Current optical technologies impose limitations on the number of wavelengths available per fiber. Typically, the number is between 20 and 100 per fiber. When the number of wavelengths allowed in the network is fixed, it might be impossible to route the network to serve all hyperedges. In this case, one may try to route the network to serve as many hyperedges as possible. This is why we study this (new) problem which seems to be the dual version of the MCHEC problem, and we call it the *Maximum Packing and Coloring Hyperedges in a Cycle* problem (MPCHC). Given a hypergraph and k colors (equivalently wavelengths), the objective is to pack and color the maximum number of hyperedges as paths in a cycle such that each physical link in the cycle is not used beyond its capacity and any two paths, each spanning all nodes of its corresponding hyperedge, must be assigned different colors (equivalently wavelengths) if they share a common physical link.

In this paper, we consider a generalization of the MPCHC problem, where different hyperedges may have different profits. When there are limited colors (equivalently wavelengths), the problem is to pack and color a subset of the hyperedges of a hypergraph as paths in a cycle. The objective is to maximize the total profit of the hyperedges selected, where each physical link e_j on the cycle is used at most c_j times, each hyperedge h_i has a profit p_i and any two paths, each spanning the nodes of its corresponding hyperedge, must be assigned different colors if they share a common physical link. We call this problem *Maximizing Profits when Packing and Coloring Hyperedges in a Cycle* (MPPCHC).

To describe the MPCHC problem in mathematical terms, we need the following terminology. Let $\mathcal{P} = \{P_1, \ldots, P_q\}$ be a set of q paths on the cycle. The set $\mathcal{P}$ is called *k-colorable* if we can assign k colors to the paths in $\mathcal{P}$ such that any two paths sharing a common physical link on the cycle must be assigned different colors. For a subset $\mathcal{Q} \subseteq \mathcal{P}$ and each physical link e_j, the *load* $L(\mathcal{Q}, e_j)$ of e_j corresponding to $\mathcal{Q}$ is the number of paths in $\mathcal{Q}$ that contains the link e_j, i.e., $L(\mathcal{Q}, e_j) = |\{P \in \mathcal{Q} : P \text{ contains the link } e_j\}|$.

The MPPCHC problem is stated as the following form:

INSTANCE: A set C of k colors, a cycle $G = (V, E)$ on n nodes, a set of m hyperedges $H = \{h_1, h_2, \ldots, h_m\}$ of a hypergraph $G' = (V, H)$, a 'capacity' function $c : E \to \mathcal{Z}^+$ and a 'profit' function $p : H \to \mathcal{R}^+$.

QUESTION: Design a routing $\mathcal{Q}$ consisting of a subset of paths on G, each of which corresponds to one of the hyperedges in H selected, such that $\mathcal{Q}$ is k-colorable, $L(\mathcal{Q}, e_j) \leqslant c_j$ holds for all $1 \leqslant j \leqslant n$ and $\sum_{P_i \in \mathcal{Q}} p_i$ is maximum.

It is easy to prove the following facts: (1) The MPCHC problem is a special case of the MPPCHC problem, where each hyperedge has the same profit 1 and the capacity c_j of each link e_j is k, and (2) The *Maximum Packing Hyperedges in a Cycle* problem (MPHC) is also a special case of the MPPCHC problem, where the k-colorable constraint is omitted, equivalently, the number of colors is equal to the number of hyperedges of the hypergraph so each of the m hyperedges is assigned to one of m colors. Because of this, the MPHC problem is the dual version of the MCHEC problem.

For a hypergraph consisting of n nodes and m hyperedges, the MCHEC problem is solvable in polynomial time with the minimum congestion c if and only if the MPHC problem is solvable in polynomial time with the optimal value m, where the capacities of links of the cycle are the same c. So the *NP*-hardness of the MCHEC problem [7] implies that the MPHC problem is *NP*-hard, which also shows that the MPPCHC problem is *NP*-hard.

In this paper, we shall first present a 2-approximation algorithm for the MPPCHC problem and then a $\frac{3}{2}$-approximation algorithm for the MPCHC problem.

The rest of the paper is organized as follows. In the section 2, we introduce some preliminaries and notation for hypergraphs and state the minimum-cost flow theory results that we use to design our approximation algorithm. In the section 3, we present a 2-approximation algorithm for the MPPCHC problem. We represent a $\frac{3}{2}$-approximation algorithm for the MPCHC problem in the section 4. We conclude our work with some remarks and discussion about future research directory in the last section.

2. Preliminaries and fundamental algorithm

An n-node cycle C is an undirected graph $G = (V, E)$ with node set $V = \{i | 1 \leqslant i \leqslant n\}$ and physical link set $E = \{e_i | 1 \leqslant i \leqslant n\}$, where each physical link e_i connects two nodes i and $i + 1$ for $i = 1, 2, \ldots, n$, we also treat the node $n + i$ as the node i for $1 \leqslant i \leqslant n$ and so on. We may consider the numbers on the nodes ordered in the clockwise direction. Let $H = (V, E_H)$ be a hypergraph with the same node set $V = \{i | 1 \leqslant i \leqslant n\}$ as C and the hyperedge set $E_H = \{h_1, h_2, \ldots, h_m\}$, where each hyperedge h_i is a subset of V with two or more nodes.

For each $1 \leqslant i \leqslant m$, a *connecting path* (simply a *c-path*) P_i for hyperedge h_i is a *minimal* path in G such that all nodes in h_i are in P_i, thus the two end-nodes of P_i must be in the hyperedge h_i. For the hyperedge $h_i = \{v_1^i, v_2^i, \ldots, v_{m_i}^i\}$, where $v_1^i, v_2^i, \ldots, v_{m_i}^i$ are successively located on the cycle G in the clockwise direction, there exist exactly m_i possible c-paths $P(v_1^i, v_{m_i}^i)$, $P(v_2^i, v_1^i)$, …, $P(v_{m_i}^i, v_{m_i-1}^i)$, where $P(v_t^i, v_{t-1}^i)$ $(1 \leqslant t \leqslant m_i)$ is the path from the node v_t^i to the node v_{t-1}^i along the clockwise direction on G. For example shown in Fig. 1, if a hyperedge $h = \{v_1, v_2, v_3, v_4\}$, we have four c-paths $P(v_1, v_4)$, $P(v_2, v_1)$, $P(v_3, v_2)$ and $P(v_4, v_3)$ on the cycle, and such a c-path spans four nodes v_1, v_2, v_3, v_4.

2.1. The MPC problem

To design an approximation algorithm for the MPPCHC problem, our basic idea is to delete a suitable physical link, say $e_n = (n, 1)$, from the cycle G to get the chain L that consists of the same nodes as G. By restricting the same hyperedges to the chain L, we design an optimal algorithm on the chain L for the following MPC problem described. By combining additional greedy method, we then derive a 2-approximation algorithm to the MPPCHC problem.

Our *maximizing profits in chain* (MPC) problem is stated as follows:

INSTANCE: A set C of k colors, a chain $L = (V, E)$ consisting of n nodes, a set of m paths $\mathcal{P} = \{P_1, \ldots, P_m\}$, a 'capacity' function $c : E \to \mathcal{Z}^+$ and a 'profit' function $p : \mathcal{P} \to \mathcal{R}^+$.

QUESTION: Find a subset $\mathcal{Q} \subseteq \mathcal{P}$ such that $\mathcal{Q}$ is k-colorable, $L(\mathcal{Q}, e_j) \leqslant c_j$ holds for all $1 \leqslant j < n$ and $\sum_{P_i \in \mathcal{Q}} p_i$ is maximized.

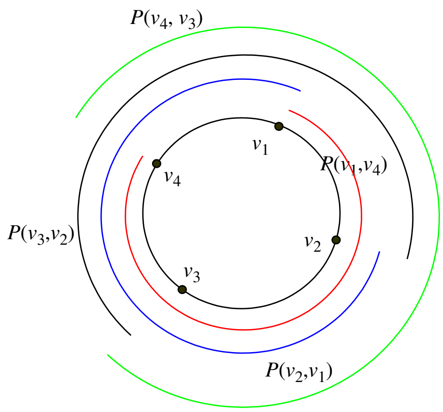

Fig. 1. Four paths span the four nodes v_1, v_2, v_3, v_4 on the cycle.

2.2. Minimum-cost flow problem

To solve the MPC problem optimally in polynomial time, our approach is to transform the MPC problem to the *minimum-cost s–t flow* problem. Edmonds and Karp [5] first showed that the minimum-cost *s–t* flow problem is solvable in strongly polynomial time, and there are many polynomial algorithms to solve the minimum-cost *s–t* flow problem [16].

Here, we need the following result due to Tarjan to solve minimum-cost *s–t* flow problem.

Theorem 1 (*Tarjan [18]*). *If a digraph $D=(V, A; s, t; c, p)$ is acyclic and capacities are all integers, then an integer minimum-cost s–t flow of value k can be found in time $\mathcal{O}(|A|+kS(n))$, even if negative costs are permitted, where $S(n)$ is the running time of any algorithm for finding shortest paths in graphs with $\mathcal{O}(n)$ edges.*

2.3. An optimal algorithm to the MPC problem

For $\mathcal{P}=\{P_1, \ldots, P_m\}$ in the instance $\mathcal{I}$ of the MPC problem on the chain L, where P_i has two end-nodes s_i and t_i $(s_i<t_i)$ for $1\leqslant i<m$, let $x_j=L(\mathcal{P}, e_j)$ for each link e_j $(1\leqslant j<n)$, and $p_{\max}=\max\{p_i : 1\leqslant i\leqslant m\}$.

The new acyclic directed network is constructed as $N=(V, A; s, t; c, p)$, where V contains n nodes labelled as $1, 2, \ldots,$ and n, the source s is the first node 1 in V and the sink t is the last node n, and the arc set A, capacity function c and cost function p are defined in the following three cases, respectively:

Case 1: For $j=1, 2, \ldots, n-1$, construct a 'clique-arc' $e_j=(j, j+1)$ in A with capacity k and cost *zero*, i.e., $c(e_j)=k$ and $p(e_j)=0$ for such a 'clique-arc' $e_j=(j, j+1)$.

Case 2: For $i=1, 2, \ldots, m$, construct an 'interval-arc' $e_i^*=(s_i, t_i)$ in A with capacity 1 and cost $-p_i$, i.e., $c(e_i^*)=1$ and $p(e_i^*)=-p_i$ for such an 'interval-arc' $e_i^*=(s_i, t_i)$.

Case 3: For $j=1, 2, \ldots, n-1$, if $\min\{k, x_j\}>c_j$, construct a 'dummy arc' $e_j^{**}=(j, j+1)$ in A with capacity $\min\{k, x_j\}-c_j$ and cost $-p_{\max}m-1$, i.e., $c(e_j^{**})=\min\{k, x_j\}-c_j$ and $p(e_j^{**})=-p_{\max}m-1$ for such a 'dummy arc' $e_j^{**}=(j, j+1)$.

Fig. 2 gives an example to construct such an acyclic network. For the chain L of 12 nodes, a set of paths is $\{P_1, P_2, P_3, P_4, P_5\}$, $k\geqslant 3$ and $c(e)=2$ for each e on L, then five arcs (1, 5), (3, 8), (4, 8), (6, 11), (9, 12) are 'interval-arc's, and three arcs (4, 5), (6, 7), (7, 8) are 'dummy arc's.

For a chain L of order n and m paths of the instance $\mathcal{I}$ of the MPC problem on the chain L, our new acyclic directed network $N=(V, A; s, t; c, p)$ contains n nodes and $n+m+\sum_{j=1}^{n-1}\max\{0, |\{j:\min\{k, x_j\}>c_j\}|\}\leqslant 2n+m$ arcs.

We need the following result to solve the MPC problem optimally from [15]. (The full proof can be found in the Appendix, too.)

Lemma 2 (*Li et al. [15]*). *Let $p(OPT)$ be the value of an optimal solution to an instance $\mathcal{I}$ of the MPC problem and $N=(V, A; s, t; c, p)$ be the directed acyclic network constructed from $\mathcal{I}$. The cost value $c(f)$ of the minimum-cost*

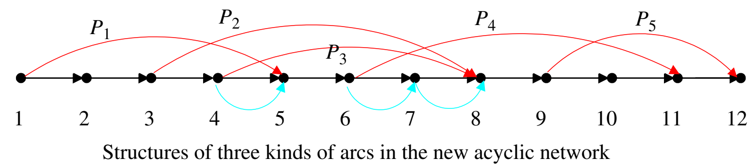

Fig. 2. The construction of the new network.

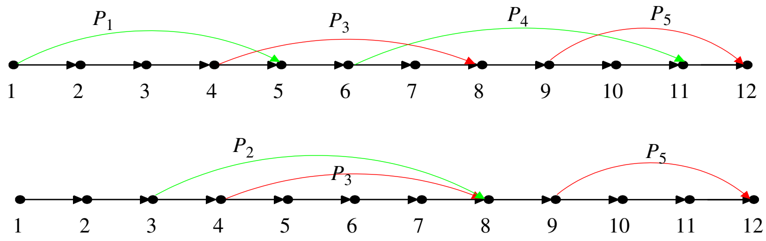

Fig. 3. Two optimal solutions to the preceding example.

s–t integer flow f of value k in $N = (V, A; s, t; c, p)$ *is*

$$c(f) = -p(OPT) - (p_{\max} m + 1) \sum_{j=1}^{n-1} \max\{\min\{k, x_j\} - c_j, 0\}.$$

Moreover, given an optimal solution for the MPC *problem, we can construct a minimum-cost s–t integer flow of value k in* $N = (V, A; s, t; c, p)$ *in polynomial time and vice versa.*

Using Tarjan's algorithm for minimum-cost *s–t* integer flow problem, we can solve the MPC problem in polynomial time. The complete algorithm is designed as follows:

Algorithm. Maximizing Profits in Chains (MPC)
INPUT: A set C of k colors, a chain $G = (V, E)$, a set of m paths $\mathscr{P} = \{P_1, \dots, P_m\}$, a 'capacity' function $c : E \to \mathscr{Z}^+$ and a 'profit' function $p : \mathscr{P} \to \mathscr{R}^+$;
OUTPUT: Find a subset $\mathscr{Q} \subseteq \mathscr{P}$ to satisfy the k-feasibility such that the objective $\sum_{P_i \in \mathscr{Q}} p_i$ is maximum.
Step 1: Construct the acyclic directed network $N = (V, A; s, t; c, p)$.
Step 2: Using Tarjan's algorithm, compute the minimum-cost *s–t* integer flow for $N = (V, A; s, t; c, p)$.
Step 3: Construct the set $\mathscr{Q}$ of 'interval-arc's of value 1, i.e., $\mathscr{Q} = \{e_i^* = (s_i, t_i) : f(e_i^*) = 1, 1 \leqslant i \leqslant m\}$.
Step 4: For all 'interval-arc's in $\mathscr{Q}$, assign the same color to the arcs that belong to the identical path sharing a unit of flow from the source *s* to the sink *t*.
(There are at most *k* colors used since the integer flow *f* has its value *k*.)
End of MPC
Thus, we can solve the MPC problem optimally from [15]. (The proof can be found in the appendix, too.)

Theorem 3 (*Li et al.* [15]). *Algorithm* MPC *solves the* MPC *problem in time* $\mathcal{O}(k(2n + m))$, *where n is the number of nodes, m is the number of paths on the chain and k is the number of colors.*

For example shown in Fig. 2, when the paths P_1, P_2, P_3, P_4 and P_5 have the profits 5, 7, 10, 2 and 4, respectively, by utilizing the algorithm MPC, we can obtain the optimal solution consisting of the paths P_1, P_3, P_4 and P_5 with the optimal value 21. See Fig. 3(a). It is easy to see that another optimal solution consists of the paths P_2, P_3 and P_5 with the same optimal value 21. See Fig. 3(b).

3. A 2-approximation algorithm for the MPPCHC problem

In this section, we study the MPPCHC problem. For each hyperedge $h_i = \{v_1^i, v_2^i, \ldots, v_{m_i}^i\}$, the main difficulty is that there exist m_i $(=|h_i|)$ choices to serve the hyperedge h_i in the cycle G. The basic ideas to solve the problem are: (i) delete a suitable link e_j with the minimum capacity from G to construct the chain L, (ii) utilize the algorithm MPC for L to obtain an optimal solution to the MPC problem fixed on L, (iii) utilize the greedy method to choose a set of $\min\{k, c_j\}$ suitable c-paths passing through e_j with the heaviest profits, each c-path being corresponding to its hyperedge, and (iv) select the better solution obtained in (ii) and (iii). Thus we can design a 2-approximation algorithm.

Without loss of generality, we may assume that the physical link $e_n = (n, 1)$ has the minimum capacity on G. Let L be the chain obtained by deleting e_n from G. Then L has the same nodes as G. For each hyperedge h_i of a hypergraph, *different* from the facts that there exist $|h_i|$ c-paths on G to span all nodes in h_i, there exists a *unique* c-path on L to span all nodes in h_i on L, then we choose such a unique c-path on L to span all nodes in h_i (this c-path is also on G, but it does not pass through e_n in G).

For each hyperedge $h_i = \{v_1^i, v_2^i, \ldots, v_{m_i}^i\}$, we may assume $1 \leqslant v_1^i < v_2^i < \cdots < v_{m_i}^i \leqslant n$. Since h_i determines a unique c-path $P_i = P(v_1^i, v_{m_i}^i)$ on the chain L to span all nodes in h_i such that P_i has two end-nodes v_1^i and $v_{m_i}^i$ on L, we may *exchangeably* use P_i and h_i to represent the same matter on L.

Our algorithm for the MPPCHC problem is described as the following form.

Algorithm. Maximum Profits when Packing and Coloring Hyperedges in a Cycle (MPPC)
INPUT: An instance $\mathscr{I}$ of the MPPCHC problem.
OUTPUT: A feasible set of paths $\mathscr{Q}$, corresponding to the set of hyperedges H', such that $\sum_{P_j \in \mathscr{Q}} p_j$ is maximized as possible.

Step 1: Choose a link $e_n = (n, 1)$ to satisfy $c_n = \min\{c_j | 1 \leqslant j \leqslant n\}$.

Step 2: Delete the link $e_n = (n, 1)$ from the cycle G, and then obtain the chain L.

Step 3: Denote $D = \{\{v_1^i, v_{m_i}^i\}: h_i = \{v_1^i, v_2^i, \ldots, v_{m_i}^i\}, 1 \leqslant i \leqslant m\}$.

Step 4: Use the algorithm MPC on L with the pairs in D, the source $s = 1$ and the sink $t = n$. Then let D' be the optimal set obtained (note $D' \subseteq D$).

Step 5: Denote $H' = \{h_i = \{v_1^i, v_2^i, \ldots, v_{m_i}^i\}\}: \{v_1^i, v_{m_i}^i\} \in D'\}$.

Step 6: Let $\mathscr{P}$ be the set of c-paths on L according to the pairs in D' as well as corresponding to the hyperedges in H', then each c-path P_i in $\mathscr{P}$ has the two end-nodes v_1^i and $v_{m_i}^i$ for some $\{v_1^i, v_{m_i}^i\} \in D'$ as well as the hyperedge $h_i = \{v_1^i, v_2^i, \ldots, v_{m_i}^i\} \in H'$, and then each pair $\{v_1^i, v_{m_i}^i\} \in D'$ uniquely determines a c-path P_i in $\mathscr{P}$.

Step 7: By using the greedy method, choose $\min\{k, c_n\}$ hyperedges in H to have the heaviest profits, and then use $\mathscr{Q}$ to represents the set of such $\min\{k, c_n\}$ c-paths, each containing the link e_n to route. And denote H'' to represent the set of such $\min\{k, c_n\}$ hyperedges in H.

Step 8: Output the better solution of $\mathscr{P}$ and $\mathscr{Q}$ corresponding its set of hyperedges (either H' or H'').

End of MPPC

We have the four facts from the MPPC algorithm: (i) for all 'interval-arc's in $\mathscr{P}$, we can assign the same color to the arcs that belong to the identical path sharing a unit of flow from the source 1 to the sink n, (ii) since the link e_n has the minimum capacity on G, we can choose $\min\{k, c_n\}$ hyperedges from H such that no link capacity is violated when we pack such $\min\{k, c_n\}$ hyperedges selected as paths in G, (iii) for each hyperedge selected h_i consisting of $|h_i|$ nodes at the step 7, we can choose any path corresponding to its hyperedge from the $|h_i| - 1$ c-paths that pass through the link e_n to span all $|h_i|$ nodes in h_i, and (iv) the $\min\{k, c_n\}$ paths in $\mathscr{Q}$ (corresponding to hyperedges), which pass through the link e_n, can be assigned different colors.

Now, we obtain the main result to the MPPCHC problem

Theorem 4. *The algorithm* MPPC *is a* 2*-approximation algorithm to solve the* MPPCHC *problem and it runs in time* $\mathcal{O}(k(2n + m))$, *where n is the number of nodes of the cycle G*, *m is the number of hyperedges of a hypergraph and k is the number of colors.*

Proof. For an optimal set *OPT* of c-paths and an output set *OUT* of c-paths to the MPPCHC problem, respectively, denote $p(OPT)$ and $p(OUT)$ to be the values of the sets *OPT* and *OUT*, respectively. Then $p(OUT) =$

$\max\{\sum_{P_i \in \mathscr{P}} p_i, \sum_{P_j \in \mathscr{Q}} p_j\}$ from the MPPC algorithm. Let OPT_1 be the set of c-paths in *OPT* that do not pass through the link e_n and OPT_2 the set of c-paths in *OPT* that pass through the link e_n. So *OPT* can be partitioned into two subsets OPT_1 and OPT_2, i.e., $OPT = OPT_1 \cup OPT_2$.

From Theorem 3, we get $p(OPT_1) \leqslant \sum_{P_i \in \mathscr{P}} p_i \leqslant \max\{\sum_{P_i \in \mathscr{P}} p_i, \sum_{P_j \in \mathscr{Q}} p_j\}$. So we derive $p(OPT_1) \leqslant p(OUT)$.

Now, we want to show that the greedy method at the step 7 ensures $p(OPT_2) \leqslant \sum_{h_j \in \mathscr{Q}} p_j \leqslant p(OUT)$. Since the link e_n has the minimum capacity on G, the $\min\{k, c_n\}$ hyperedges selected at the step 7 in the algorithm satisfies the property: when these $\min\{k, c_n\}$ hyperedges selected as paths are packed in G, no link capacity is violated. Denote $\mathscr{Q} = \{P_{j_1}, \ldots, P_{j_{\min\{k,c_n\}}}\}$ and $OPT_2 = \{P_{j'_1}, \ldots, P_{j'_{m'}}\}$. Without loss of generality, we may assume that the weights satisfy: $p_{j_1} \geqslant p_{j_2} \geqslant \cdots \geqslant p_{j_{\min\{k,c_n\}}}$ and $p_{j'_1} \geqslant p_{j'_2} \geqslant \cdots \geqslant p_{j'_{m'}}$. Since OPT_2 is a feasible solution to an instance $\mathscr{I}$ of the MPPCHC problem, we get $\min\{k, c_n\} \geqslant m'$. By the fact that we choose the $\min\{k, c_n\}$ hyperedges corresponding to the c-paths with the heaviest profits by the greedy method at the step 7, we obtain $p_{j'_l} \leqslant p_{j_l}$ for each $1 \leqslant l \leqslant m'$ ($\leqslant \min\{k, c_n\}$).Then $p(OPT_2) = \sum_{l=1}^{m'} p_{j'_l} \leqslant \sum_{l=1}^{m'} p_{j_l} \leqslant \sum_{l=1}^{\min\{k,c_n\}} p_{j_l} = \sum_{h_{j_l} \in \mathscr{Q}} p_{j_l} \leqslant \max\{\sum_{P_i \in \mathscr{P}} p_i, \sum_{P_j \in \mathscr{Q}} p_j\}$. Thus we get $p(OPT_2) \leqslant p(OUT)$.

Hence, we conclude $p(OPT) = p(OPT_1) + p(OPT_2) \leqslant 2p(OUT)$, which implies that MPPC is a 2-approximation algorithm for the MPPCHC problem.

Now, we analyze the complexity of the algorithm MPPC. At the step 1, it needs n steps to choose the link with the minimum capacity from G; at the steps 4–6, Theorem 3 provides the complexity $\mathcal{O}(k(2n+m))$; at the step 7, it needs $\mathcal{O}(k)$ steps to choose the $\min\{k, c_n\}$ hyperedges to have the heaviest profits. Other steps need constant time to execute. Thus, the algorithm MPPC runs in time $\mathcal{O}(k(2n+m))$.

This completes the proof of Theorem 4. □

4. A $\frac{3}{2}$-approximation algorithm to the MPCHC problem

In this section, by combining the algorithm MPC and matching theory, we design a $\frac{3}{2}$-approximation algorithm for the MPCHC problem.

Two hyperedges $h_s = \{v_1^s, v_2^s, \ldots, v_{m_s}^s\}$ and $h_t = \{u_1^t, u_2^t, \ldots, u_{m_t}^t\}$ of a hypergraph are called *parallel* on the cycle G if there exist two c-paths P_{h_s} and P_{h_t} such that two paths P_{h_s} and P_{h_t} contain no common physical link on G, where P_{h_s} spans all $|h_s|$ nodes in h_s and P_{h_t} spans all $|h_t|$ nodes in h_t, respectively. See Fig. 4(a). Otherwise, they are called *crossing* on G, i.e., for each c-path P_{h_s} spanning all $|h_s|$ nodes in h_s and each c-path P_{h_t} spanning all $|h_t|$ nodes in h_t, these two paths P_{h_s} and P_{h_t} always contain at least one common physical link on G. See Fig. 4(b). If two hyperedges h_s and h_t are parallel on G, then there exist two c-paths P_{h_s} and P_{h_t} on G, where P_{h_s} spans all $|h_s|$ nodes in h_s and P_{h_t} spans all $|h_t|$ nodes in h_t, such that both paths are edge-disjoint on G, thus they would be assigned the same color. See Fig. 4(a). Before we design a $\frac{3}{2}$-approximation algorithm to the MPCHC problem, we construct an *auxiliary* graph $D = (H, E_H)$ corresponding to the set H of hyperedges of a hypergraph, where two hyperedges h_s and h_t of H is *adjacent* in the auxiliary graph D, say $h_s h_t \in E_H$, if and only if they are parallel on G.

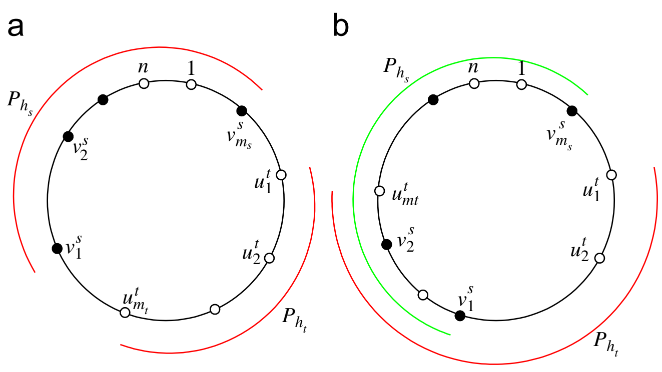

Fig. 4. Structures of two kinds of hyperedges. (a) Parallel hyperedges; (b) crossing hyperedges.

The basic ideas to solve the MPCHC problem are: (i) delete a link e_n from G to obtain the chain L_1, (ii) use the algorithm MPC for L_1 to obtain an optimal solution to the MPC problem fixed on L_1, (iii) construct an auxiliary graph, then find a maximum cardinality matching M, and choose $\min\{|M|, k\}$ pairs of parallel hyperedges, and (iv) select the better solution obtained in (ii) and (iii).

Now, we present our approximation algorithm to the MPCHC problem:

Algorithm. Maximum packing and coloring hyperedges in a cycle (MPCH)
INPUT: An instance $\mathscr{I}$ of the MPCHC problem.
OUTPUT: A feasible solution *OUT* containing as many hyperedges as possible.
Step 1: Delete the link $e_n = (n, 1)$ from the cycle G to obtain a chain L_1 with the source 1 and the sink n.
Step 2: Denote $H_2 = \{\{v_1^i, v_{m_i}^i\}\}$: a hyperedge $h_i = \{v_1^i, v_2^i, \ldots, v_{m_i}^i\}$ $(\in H)$ satisfies $v_1^i < v_2^i < \cdots < v_{m_i}^i$, $1 \leqslant i \leqslant m\}$.
Step 3: Use the algorithm MPC on L_1 with the pairs in H_2, the source $s = 1$ and the sink $t = n$. And denote H_2' to be the optimal set obtained (note $H_2' \subseteq H_2$).
Step 4: Denote $H' = \{h_i = \{v_1^i, v_2^i, \ldots, v_{m_i}^i\} \in H : \{v_1^i, v_{m_i}^i\} \in H_2'\}$.
Step 5: Let $\mathscr{P}$ be the set of c-paths on L_1 according to the pairs in H_2' (also corresponding to the hyperedges in H'), i.e., each c-path in $\mathscr{P}$ has the two end-nodes v_1^i and $v_{m_i}^i$ for some $\{v_1^i, v_{m_i}^i\} \in H_2'$, and then each pair $\{v_1^i, v_{m_i}^i\}$ $(\in H_2')$ uniquely determines a c-path in $\mathscr{P}$.
(*/ the set $\mathscr{P}$ is k-colorable and $L(\mathscr{P}, e_j) \leqslant k$ holds for all $1 \leqslant j < n$ /*)
Step 6: Construct the auxiliary graph $D = (H, E_H)$, and find a maximum cardinality matching M in D.
Step 7: Choose any $\min\{|M|, k\}$ edges from M, each corresponding to the two parallel hyperedges $\in H$; hence, these $2\min\{|M|, k\}$ parallel hyperedges, corresponding to the c-paths set $\mathscr{Q}$, form a feasible solution.
(*/ the set $\mathscr{Q}$ is also k-colorable with exactly $2\min\{|M|, k\}$ paths /*)
Step 8: Output the better solution obtained in steps 5 and 7.
End of MPCH

We show that the algorithm MTRR has the approximation ratio $\frac{3}{2}$ in the following theorem.

Theorem 5. *The algorithm* MPCH *is a* $\frac{3}{2}$*-approximation algorithm to solve the* MPCHC *problem, and it runs in time* $\mathcal{O}(k(2n+m)+M(m))$, *where* n *is the number of nodes of the cycle* G, m *is the number of hyperedges of a hypergraph*, k *is the number of colors and* $M(m)$ *is the running time to obtain a maximum cardinality matching in a graph on* m *nodes*.

Proof. For an optimal solution *OPT* (set of c-paths as well as set of hyperedges) to the MPCHC problem and the feasible solution *OUT* (set of c-paths as well as set of hyperedges) obtained from the algorithm MPCH, let OPT_1 be the set of c-paths in *OPT* that do not pass through the physical link e_n and OPT_2 the set of c-paths in *OPT* that pass through the physical link e_n. So *OPT* can be partitioned into two subsets OPT_1 and OPT_2, i.e., $OPT = OPT_1 \cup OPT_2$. Denote C_2 to represent the set of colors that are assigned to the c-paths in OPT_2. Then $|OPT_2| = |C_2| \leqslant k$ by the definition of the set OPT_2.

We may choose an optimal solution *OPT* such that C_2 contains the least number of colors among all optimal solutions to an instance $\mathscr{I}$ of the MPCHC problem, equivalently, there are minimum $|C_2|$ colors assigned to the c-paths in OPT_2. Then each color in C_2 must be assigned to some c-paths in OPT_1. (Otherwise, if a color c in C_2 is not assigned to any c-paths in OPT_1, we choose the c-path $P_{h_s}(v_1^s, v_{m_s}^s)$ from OPT_2 assigned by such a color c, corresponding to the hyperedge $h_s = \{v_1^s, \ldots, v_r^s, v_{r+1}^s, \ldots, v_{m_s}^s\}$, where $1 \leqslant v_{r+1}^s < \cdots < v_{m_s}^s < \cdots < v_1^s < \cdots < v_r^s \leqslant n$. See Fig. 5(a). Noting that $P_{h_s}(v_1^s, v_{m_s}^s)$ passes through e_n along the clockwise direction to span all $|h_s|$ nodes in h_s from v_1^s to $v_{m_s}^s$, we can construct another c-path $P_{h_s}(v_{r+1}^s, v_r^s)$ to span all $|h_s|$ nodes in h_s from v_{r+1}^s to v_r^s along clockwise direction, then $P_{h_s}(v_{r+1}^s, v_r^s)$ does not pass through e_n and it is assigned the same color c. See Fig. 5(b). Thus we obtain a new optimal solution whose the number of paths through e_n is exactly $|C_2| - 1$, contradicting the choice of the optimal solution *OPT*.)

By the fact that each color in C_2 must be assigned to some c-paths in OPT_1, we get the fact that each color in C_2 must be assigned to a pair of c-paths, where one c-path is in OPT_1 and another is in OPT_2. Then these corresponding hyperedges are adjacent in the auxiliary graph D. Such $|C_2|$ pairs of hyperedges form a matching in D. By the

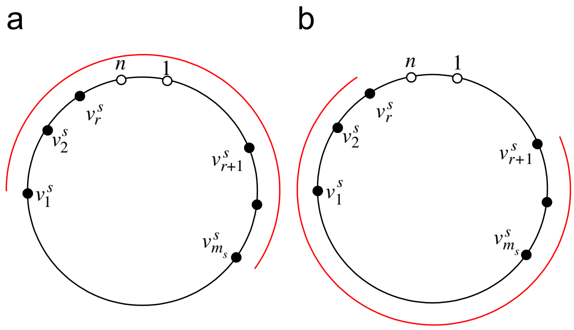

Fig. 5. Construction of P'_{h_s} from P_{h_s}. (a) The path $P_{h_s}(v_1^s, v_{m_s}^s)$; (b) the path $P_{h_s}(v_{r+1}^s, v_r^s)$

algorithm MPCH, M is a maximum cardinality matching in D. Thus, $|C_2| \leqslant |M|$. Again, since each path in OPT_2 passes through e_n, we obtain $|C_2| \leqslant k$. Hence, we obtain $|OPT_2| = |C_2| \leqslant \min\{|M|, k\}$.

On the other hand, Theorem 3 ensures that the algorithm MPCH implies $|\mathscr{P}| \geqslant |OPT_1| = |OPT| - |OPT_2|$, then we obtain $|OPT| \leqslant |\mathscr{P}| + |OPT_2| \leqslant |\mathscr{P}| + \min\{|M|, k\}$.

Step 7 in the algorithm MPCH implies $2\min\{|M|, k\} = |\mathscr{Q}|$, i.e., $\min\{|M|, k\} = |\mathscr{Q}|/2$. Hence, we conclude

$$|OPT| \leqslant |\mathscr{P}| + \min\{|M|, k\} = |\mathscr{P}| + \frac{|\mathscr{Q}|}{2} \leqslant \frac{3}{2}|OUT|.$$

We analyze the complexity of algorithm MPCH. At the step 3, it needs $\mathcal{O}(k(2n+m))$ steps to obtain the optimal set H'_2 by Theorem 3; at the step 6, it needs $M(m)$ steps to find a maximum cardinality matching in the auxiliary graph D; and the other steps need time $\mathcal{O}(m)$ to execute. Thus, the algorithm MPPC runs in time $\mathcal{O}(k(2n+m) + M(m))$.

This completes the proof of Theorem 5. □

5. Conclusion and further work

In this paper, we study the problem of packing and coloring some hyperedges of a hypergraph in a cycle so as to maximize profits of the different paths selected, where no link capacity is violated and two c-paths (and the corresponding hyperedges) must be assigned different colors when they have a common link. We derive a 2-approximation algorithm to the MPPCHC problem and then a $\frac{3}{2}$-approximation algorithm to the MPCHC problem.

Since the MCHEC problem admits a PTAS [4], can we design a PTAS or better approximation algorithms with approximation ratio less than $\frac{3}{2}$ to the MPCHC problem? Can we find other better approximation algorithms for the MPPCHC problem with approximation ratio less than two? These problems are very challenging and interesting to be discussed in further study.

Acknowledgments

The authors are very grateful to an anonymous referee whose helpful comments and suggestions have led to a substantially improved presentation of the paper.

Jianping Li is fully supported by the National Natural Science Foundation of China [nos. 10561009, 10271103] and Natural Science Foundation of Yunnan Province [no. 2006F0016M].

Lusheng Wang and Hao Zhao are fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China [no. Cityu 1196/03E].

Appendix 1. Proofs of Lemma 2 and Theorem 3

Lemma 2. *Let $p(OPT)$ be the value of an optimal solution to an instance $\mathscr{I}$ of the* MPC *problem and $N = (V, A; s, t; c, p)$ be the directed acyclic network constructed from $\mathscr{I}$. The cost value $c(f)$ of the minimum-cost s–t* *integer flow f of value k in* $N=(V,A;s,t;c,p)$ *is*

$$c(f)=-p(OPT)-(p_{\max}m+1)\sum_{j=1}^{n-1}\max\{\min\{k,x_j\}-c_j,0\}.$$

Moreover, given an optimal solution for the MPC *problem, we can construct a minimum-cost s–t integer flow of value k in* $N=(V,A;s,t;c,p)$ *in polynomial time and vice versa.*

Proof. Suppose that we have an optimal solution *OPT* with profit $p(OPT)$ to an instance $\mathscr{I}$ of the MPC problem, i.e., $\mathscr{P}_{\rm OPT}=\{P_{s_{i_1},t_{i_1}},\ldots,P_{s_{i_r},t_{i_r}}\}$ is an optimal solution to an instance $\mathscr{I}$ of the MPC problem such that $p(\mathrm{OPT})=\sum_{(s_i,t_i)\in P_{\rm OPT}}p_i$. Denote $P_{\rm OPT}=\{(s_{i_1},t_{i_1}),\ldots,(s_{i_r},t_{i_r})\}$. Let $\mathscr{P}=\{P_1,\ldots,P_m\}$ be the set of paths, corresponding to the set of *m* requests. For each $1\leqslant j<n$, denote $x_j=L(\mathscr{P},e_j)$ and $z_j=L(\mathscr{P}_{\rm OPT},e_j)$.

Now, we can construct a function $f:A\to\mathscr{R}^+$ as follows:

(i) For an 'interval-arc' $e_i^*=(s_i,t_i)$ $(i\in\{1,2,\ldots,m\})$, if $e_i^*\in P_{\rm OPT}$, define $f(e_i^*)=1$, and if $e_i^*\notin P_{OPT}$, define $f(e_i^*)=0$.
(ii) For a 'dummy-arc' $e_j^{**}=(j,j+1)$ $(j\in\{1,2,\ldots,n-1\})$, by construction, $\min\{k,x_j\}>c_j$, define $f(e_j^{**})=\min\{k,x_j\}-c_j$.
(iii) For a 'clique-arc' $e_j=(j,j+1)$ $(j\in\{1,2,\ldots,n-1\})$, if $\min\{k,x_j\}>c_j$, define $f(e_j)=k-(\min\{k,x_j\}-c_j)-z_j$, and if $\min\{k,x_j\}\leqslant c_j$, define $f(e_j)=k-z_j$.

It is easy to check that *f* is an *s–t* integer flow of value *k* and its cost value is

$$\begin{aligned}
cost(f)&=\sum_{e\in A}p(e)f(e)\\
&=\sum_{i=1}^{m}p(s_i,t_i)f(s_i,t_i)+\sum_{e_j^{**}:\min\{k,x_j\}>c_j}p(e_j^{**})f(e_j^{**})+\sum_{j=1}^{n-1}p(e_j)f(e_j)\\
&=\sum_{e_i^*\in P_{\rm OPT}}p(e_i^*)f(e_i^*)+\sum_{e_j^{**}:\min\{k,x_j\}>c_j}p(e_j^{**})f(e_j^{**})+\sum_{j=1}^{n-1}0\cdot f(e_j)\\
&=\sum_{e_i^*\in P_{\rm OPT}}(-p_i)+\sum_{e_j^{**}:\min\{k,x_j\}>c_j}(-p_{\max}m-1)(\min\{k,x_j\}-c_j)\\
&=-\sum_{e_i^*\in P_{\rm OPT}}p_i+(-p_{\max}m-1)\sum_{e_j^{**}:\min\{k,x_j\}>c_j}(\min\{k,x_j\}-c_j)+0\\
&=-p(OPT)-(p_{\max}m+1)\sum_{j=1}^{n-1}\max\{\min\{k,x_j\}-c_j,0\},
\end{aligned}$$

where the first and second equalities depend on the definition of $cost(f)$, the third equality comes from the facts $f(e_i^*)=0$ for $e_i^*\notin P_{\rm OPT}$ and $p(e_j)=0$ for $e_j=(j,j+1)$, the fourth equality is from the facts $p(e_i^*)=-p_i$, $f(e_i^*)=1$ for $e_i^*\in P_{\rm OPT}$ and $p(e_j^{**})=-p_{\max}m-1$, $f(e_j^{**})=\min\{k,x_j\}-c_j$ for e_j^{**}. Since the second item in the last equality is a constant, the optimal profit $p(OPT)$ to an instance $\mathscr{I}$ of the MPC problem implies that the cost value $cost(f)$ of flow *f* of value *k* is smallest. So *f* is a minimum-cost *s–t* integer flow of value *k* in *N*.

Conversely, if we have a minimum-cost *s–t* integer flow *f* on *N*, since the cost per unit of each 'dummy-arc' $e_j^{**}=(j,j+1)$ in the network *N* is $-p_{\max}m-1$, then each 'dummy-arc' e_j^{**} must be chosen in the minimum-cost *s–t* integer flow f to possess its flow value $\max\{\min\{k, x_j\} - c_j, 0\}$ through that 'dummy-arc' by algorithm MPC, and the set of 'interval-arc's are partitioned into two subsets: one consists of 'interval-arc's of value *zero* and the other consists of 'interval-arc's of value *one*, which imply that the latter 'interval-arc's in the flow f consist of the arc set $P_{\mathrm{OUT}} = \{(s_{i_1}, t_{i_1}), \ldots, (s_{i_r}, t_{i_r})\}$ (and corresponding to the set of paths). So the cost value of minimum-cost s–t integer flow f of value k is as follows:

$$cost(f) = -\sum_{l=1}^{r} p_{i_l} - (p_{\max} m + 1) \sum_{j=1}^{n-1} \max\{\min\{k, x_j\} - c_j, 0\}.$$

For each arc (s_{i_l}, t_{i_l}) in P_{OUT}, we can construct a path $P_{s_{i_l}, t_{i_l}}$ to connect s_{i_l} and t_{i_l} and denote $\mathscr{P}_{\mathrm{OUT}} = \{P_{s_{i_1}, t_{i_1}}, \ldots, P_{s_{i_r}, t_{i_r}}\}$. The paths in $\mathscr{P}_{\mathrm{OUT}}$, whose corresponding 'interval-arc's belong to the identical path sharing a unit of flow from the source s to the sink t, are assigned the same color, then the paths in $\mathscr{P}_{\mathrm{OUT}}$ can be checked to be k-colorable in time $\mathscr{O}(m)$ [1] and the profit of $\mathscr{P}_{\mathrm{OUT}}$ is $p(OUT) = \sum_{l=1}^{r} p_{i_l}$. Hence we can obtain

$$\begin{aligned} p(OUT) &= \sum_{l=1}^{r} p_{i_l} \\ &= -cost(f) - (p_{\max} m + 1) \sum_{j=1}^{n-1} \max\{\min\{k, x_j\} - c_j, 0\}. \end{aligned}$$

Since the second item in the last equality is a constant, the cost value $cost(f)$ of minimum-cost s–t integer flow f of value k implies that the set of paths $\mathscr{P}_{\mathrm{OUT}}$ is an optimal solution to the instance $\mathscr{I}$ of the MPC problem. □

Theorem 3. *Algorithm* MPC *solves the* MPC *problem in* $\mathscr{O}(k(2n + m))$ *time*, *where n is the number of nodes*, *m is the number of hyperedges and k is the number of colors*.

Proof. Lemma 2 implies the correctness of algorithm MPC to solve the MPC problem. Now, we analyze the complexity of algorithm MPC.

At the step 1, it needs $\mathscr{O}(2n + m)$ time. The acyclic network N consists of n nodes and at most $2n + m$ arcs. At the step 2, we can construct a minimum-cost s–t integer flow f of value k on the acyclic network N in $\mathscr{O}(k(2n + m))$ time. At the step 3, the construction of the set of 'interval-arc's with flow value *one* needs $\mathscr{O}(m)$ time. At the step 4, the set $\mathscr{P}$ is checked to be k-colorable in $\mathscr{O}(m)$ time. Thus, the algorithm MPC runs in $\mathscr{O}(n+m) + \mathscr{O}(k(2n+m)) + \mathscr{O}(m) + \mathscr{O}(m)$ time, i.e., $\mathscr{O}(k(2n + m))$ time. □

Appendix 2. Some remark for the choice of the link e_n

Without loss of generality, we may assume that the physical link $e_n = (n, 1)$ has the minimum capacity on G, i.e., $c_n = \min\{c_j \mid 1 \leqslant j \leqslant n\}$, otherwise if the link $e_{j_0} = (j_0, j_0 + 1)$ has the minimum capacity on G, i.e., $c_{j_0} = \min\{c_j \mid 1 \leqslant j \leqslant n\}$, we can obtain a permutation

$$\sigma = \begin{pmatrix} 1 & 2 & \cdots & n - j_0 & n - j_0 + 1 & n - j_0 + 2 & \cdots & n \\ j_0 + 1 & j_0 + 2 & \cdots & n & 1 & 2 & \cdots & j_0 \end{pmatrix},$$

where we treat the node $n + i$ as the node i for $1 \leqslant i \leqslant n$ and so on, then we obtain that the link $e_{\sigma(n)} = (\sigma(n), \sigma(n) + 1)$ has the minimum capacity on G. We do the following operations under the constraints where each integer j is replaced by the integer $\sigma(j)$, then we shall obtain the final results under the constraints where each integer j' is replaced by the integer $\sigma^{-1}(j')$, where σ^{-1} is the inverse permutation of σ, i.e.,

$$\sigma^{-1} = \begin{pmatrix} 1 & 2 & \cdots & j_0 & j_0 + 1 & j_0 + 2 & \cdots & n \\ n - j_0 + 1 & n - j_0 + 2 & \cdots & n & 1 & 2 & \cdots & n - j_0 \end{pmatrix}.$$

References

[1] M. Carlisle, L. Lloyd, On the k-coloring of intervals, Discrete Appl. Math. 59 (3) (1995) 225–235.

[2] T. Carpenter, S. Cosares, I. Saniee, Demand routing and slotting on ring networks, Technical Report 97-02, Bellcore, Morristown, NJ, 1997.

[3] C. Cheng, Improved approximation algorithms for the demand routing and slotting problem with unit demands on rings, SIAM J. Discrete Math. 17 (3) (2004) 384–402.

[4] X. Deng, G. Li, A PTAS for embedding hypergraph in a cycle (extended abstract), in: Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP 2004), Turku, Finland, 2004, pp. 433–444.

[5] J. Edmonds, R. Karp, Theoretical improvements in algorithmic efficiency for network flow problems, J. Assoc. Comput. Mach. 19 (1972) 248–264.

[6] A. Frank, T. Nishizeki, N. Saito, H. Suzuki, E. Tardos, Algorithms for routing around a rectangle, Discrete Appl. Math. 40 (1992) 363–378.

[7] J. Ganley, J. Cohoon, Minimum-congestion hypergraph embedding in a cycle, IEEE Trans. Comput. 46 (5) (1997) 600–602.

[8] T. Gonzalez, Improved approximation algorithms for embedding hyperedges in a cycle, Inform. Process. Lett. 67 (1998) 267–271.

[9] Q. Gu, Y. Wang, Efficient algorithms for minimum congestion hypergraph embedding in a cycle, IEEE Trans. Parallel Distrib. Systems 17 (3) (2006) 205–214.

[10] H. Ho, S. Lee, A $1.5+\varepsilon$ approximation algorithm for weighted hypergraph embedding in a cycle, in: The 23rd Workshop on Combinatorial Mathematics and Computation Theory, pp. 16–23.

[11] S. Khanna, A polynomial time approximation scheme for the SONET ring loading problem, Bell Labs Tech. J. 2 (1997) 36–41.

[12] V. Kumar, Approximating circular arc coloring and bandwidth allocation in all-optical ring networks, in: Proceedings of the International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX'98), Lecture Notes in Computer Science, vol. 1444, 1998, pp. 147–158.

[13] S. Lee, H. Ho, On minimizing the maximum congestion for weighted hypergraph embedding in a cycle, Inform. Process. Lett. 87 (2003) 271–275.

[14] S. Lee, H. Ho, A 1.5 approximation algorithm for embedding hypergraph in a cycle, IEEE Trans. Parallel Distrib. Systems 16 (6) (2005) 481–488.

[15] J. Li, L. Wang, H. Zhao, Maximizing profits of requests in WDM networks, J. Combin. Optim. 10 (2005) 99–111.

[16] A. Schrijver, Combinatorial Optimization: Polyhedra and Efficiency, Springer, The Netherlands, 2003.

[17] A. Schrijver, P. Seymour, P. Winkler, The ring loading problem, SIAM J. Discrete Math. 11 (1) (1998) 1–14.

[18] R. Tarjan, Data Structures and Network Algorithms, SIAM, Philadelphia, PA, 1983.