國立交通大學

應用數學系

碩士論文

一個只花 *O*(log *N*)時間找出具有 *N* 點之雙環式網路的 steps 的演算法以及 Hyper-L$_1$ 三環式網路的存在性的探討

An *O*(log *N*)-Time Algorithm to Find the Steps of a Double-Loop Network with *N* Nodes and the Existence of Hyper-L$_1$ Triple-Loop Networks

研究生：唐文祥

指導老師：陳秋媛 教授

中華民國九十三年六月

一個只花 $O(\log N)$時間找出具有 N 點之 雙環式網路的 steps 的演算法以及 Hyper-L_1 三環式網路的存在性的探討

An $O(\log N)$-Time Algorithm to Find the Steps of a Double-Loop Network with N Nodes and the Existence of Hyper-L_1 Triple-Loop Networks

研究生：唐文祥　　Student：Wen-Shiang Tang

指導老師：陳秋媛 教授　　Advisor：Dr. Chiuyuan Chen

國立交通大學
應用數學系
碩士論文

A Thesis
Submitted to Department of Applied Mathematics
College of Science
National Chiao Tung University
In partial Fulfillment of Requirement
For the Degree of Master
In
Applied Mathematics
June 2004
Hsinchu, Taiwan, Republic of China

中華民國九十三年六月

一個只花 $O(\log N)$時間找出具有 N 點之雙環式網路的 steps 的演算法以及 Hyper-L_1 三環式網路的存在性的探討

研究生：唐文祥　　　指導老師：陳秋媛　教授

國立交通大學

應用數學系

摘要

雙環式網路及三環式網路是許多學者專家廣泛探討的區域網路架構。給定一個正整數 $\boldsymbol{N}$，找出具有 $\boldsymbol{N}$ 點、直徑最小的雙環式網路 DL(N;s_1,s_2)的 s_1 和 s_2（又稱為 steps）是學者專家們一直以來所想達成的目標。已知雙環式網路的 minimum distance diagram 是 L-型；對於雙環式網路而言，直徑可以很容易的由它的 L-型計算出。因此「找出具有 $\boldsymbol{N}$ 點、直徑最小的雙環式網路」的一個常見的方法是：將此問題轉換為「先找出一個直徑相當不錯的 L-型，再找出與這個 L-型對應的雙環式網路 DL(N; s_1, s_2)的 s_1 和 s_2」。給定一個雙環式網路 DL(N; s_1, s_2)，在論文[8]中，Cheng 和黃光明老師提出了一個漂亮而且只花 $O(\log N)$時間、找出對應的 L-型的演算法。但是，「給定一個 L-型，是否能夠只花 $O(\log N)$時間，找出與這個 L-型相對應的雙環式網路 DL(N; s_1, s_2)的 steps」，卻一直是一個 open problem [5]。在這篇論文裡，我們提出一個只花 $O(\log N)$時間、找出與一個給定的 L-型相對應的雙環式網路 DL(N; s_1, s_2)的 steps 的演算法。

令 $N(D)$表示一個直徑為 D 的三環式網路所能包含的最多點數。Hyper-L 型已被多位學者發現為推導出 $N(D)$的下界的一個有效的工具。然而，並非每一個 Hyper-L 型都會有一個三環式網路來得到它。截至目前為止，共有三種 hyper-L 型被學者們提出來，為了方便起見，我們分別稱它們為 hyper-L_0、hyper-L_1、hyper-L_2。在論文[7]中，陳秋媛老師、黃光明老師、李珠矽老師、以及石舜仁學長提出了 hyper-L_0 三環式網路存在的充份必要條件。在這篇論文裡，我們提出 hyper-L_1 三環式網路存在的充份必要條件。

關鍵詞：雙環式網路、L-型、直徑、演算法、三環式網路、hyper-L 型

中華民國九十三年六月

An $O(\log N)$-Time Algorithm to Find the Steps of a Double-Loop Network with N Nodes and the Existence of Hyper-L$_1$ Triple-Loop Networks

Student : Wen-Shiang Tang   Advisor : Dr. Chiuyuan Chen

Department of Applied Mathematics
National Chiao Tung University
Hsinchu 300, Taiwan, R.O.C.

Abstract

Double-loop networks and triple-loop networks have been widely studied as architecture for local area networks. Given an N, it is desirable to find a double-loop network DL(N; s_1, s_2) with its diameter being the minimum among all double-loop networks with N nodes. It is well known that the minimum distance diagram of a double-loop network yields an L-shape. Since the diameter can be easily computed from an L-shape, one method is to start with a desirable L-shape and then asks whether there exist s_1 and s_2 (also called the steps of the double-loop network) to realize it. While Cheng and Hwang [8] have given an elegant $O(\log N)$-time algorithm to find the L-shape of a double-loop network DL(N; s_1, s_2), it is an open problem whether the steps of a double-loop network with N nodes can be found in $O(\log N)$ time [5]. In this thesis, we propose an $O(\log N)$-time algorithm to find the steps of a double-loop network with N nodes.

Hyper-L tiles were proven to be an effective tool to obtain lower bounds for $N(D)$, the maximum number of nodes in a triple-loop network with diameter D. Unfortunately, not every hyper-L tile has a triple-loop network realizing it. Up to now, three types of hyper-L tiles have been proposed; for convenience, call them hyper-L$_0$, hyper-L$_1$, and hyper-L$_2$. In [7], Chen et al. derived the necessary and sufficient conditions for the existence of hyper-L$_0$ triple-loop networks. In this thesis, we shall derive the necessary and sufficient conditions for the existence of hyper-L$_1$ triple-loop networks.

Keywords: Double-loop network, L-shape, diameter, algorithm, triple-loop network, hyper-L tile

誌　　謝

兩年前機緣巧合下，來到台灣的矽谷---新竹市；幸運的在擁有『六最---最傑出研究、最精緻校園、最先進院系、最注重教學、最照顧學生、最優秀校友』的交通大學度過了我兩年的研究生涯。在兩年的研究日子裡，總覺得所獲得的遠大於比所付出的。老師們的細心教導與關愛，同學和學長的勉勵與切磋，讓我有一段充實而愉悅的研究生活，感動且溫馨的回憶。畢業在即，心中滿是感激與不捨。

首先特別要感謝我的指導教授---陳秋媛老師，老師以其豐富的學識，帶領我走入了 Algorithms、Graph theory 、Networks 等領域，在其精湛的教學演繹之下，使我們更能掌握住其中的精髓所在，在理論基礎上加深印象。除了課業上的指導之外，在生活上也受到老師相當多的照顧與關心；也總是在我徬徨無助與迷惘的時候及時給予幫助與意見；對於我的未來生涯的規劃也提供寶貴的意見，真的非常感謝老師所付出的一切。

其次要感謝黃光明老師與黃大原老師，對我的諸多幫助與關愛；也感謝傅恆霖老師及翁志文老師的教導。還有同門的建瑋、君逸，在我做研究時，不時給予我一些靈感和程式方面的協助來論證我們的推論。也感謝飛黃學長，電腦的熱情贊助，使我可以很快的觀察出一些推論的結果，對於我的生涯規劃也提供了寶貴的意見。

同時也要感謝和我的同學，正傑、嘉文、貴弘、啟賢、榮丰、喻培、抮君、致維、宏嘉、昭芳與同門的學弟及二樓研究室的學妹們。對我來說，生命中能和這些人相遇，真的是我的福氣。

最後也要感謝我的父母及家人，若不是他們長久以來的支持，不可能有我今天的小小成果。在此獻上無限的感激，謝謝你們。

紙短情長，願每位關心、照顧我的人，永遠健康快樂。

Contents

List of Figures

1 Introduction

Multi-loop networks have been widely studied as architecture for local area networks. A *multi-loop network* ML$(N;\ s_1,\ s_2,\ \cdots,\ s_t)$ has N nodes $0, 1, 2, \cdots, N-1$ and dN links, $i \rightarrow i+s_1,\ i \rightarrow i+s_2,\ \cdots,\ i \rightarrow i+s_t \pmod N$, $i = 0, 1, \cdots, N-1$. The integers $s_1, s_2, ..., s_t$ are called the *steps* of the multi-loop network. A multi-loop network is strongly connected if and only if $\gcd(N, s_1, s_2, ..., s_t) = 1$; see [4, 14, 16]. Since the literature considered only strongly connected multi-loop networks, this thesis considers only strongly connected multi-loop networks, too.

When $t = 2$, the multi-loop network is usually called the *double-loop network* and is denoted by $DL(N; s_1, s_2)$. See [4, 14, 15, 16, 18] for surveys of these networks. When $t = 3$, the multi-loop network is called the *triple-loop network* and is denoted by $TL(N; s_1, s_2, s_3)$. For details of multi-loop networks, refer to [3, 4, 15, 16].

Fiol et al. [12] proved that $DL(N; s_1, s_2)$ is strongly connected if and only if $\gcd(N,\ s_1,\ s_2) = 1$. When $DL(N; s_1, s_2)$ is strongly connected, then we can talk about a minimum distance diagram (MDD) which is a diagram with node 0 in cell $(0, 0)$, and node v in cell (i, j) if and only if $is_1 + js_2 \equiv v \pmod N$ and $i + j$ is the minimum among all (i', j') satisfying the congruence. Namely, a shortest path from 0 to v is through taking i s_1-links and j s_2-links (in any order). Note that in a cell (i, j), i is the column index and j is the row index. An MDD includes every node exactly once (in case of two shortest paths, the convention is to choose the cell with the smaller row index, i.e., the smaller j). Since $DL(N; s_1, s_2)$ is clearly node-symmetric, there is no loss of generality in assuming: node 0 is the origin of a path.

Wong and Coppersmith [19] proved that the MDD for $DL(N; s_1, s_2)$ is always an L-shape (a rectangle is considered a degeneration). An L-shape is determined by four parameters l, h, p, n as shown in Figure 1. These four parameters are the lengths of four of the six segments on the boundary of the L-shape. For example,

$DL(9;2,5)$ in Figure 2 has $l=5$, $h=3$, $p=3$, and $n=2$. Let $N=lh-pn$. Fiol et al. [12, 13] and Chen and Hwang [6] proved that there exists a $DL(N;s_1,s_2)$ realizing the L-shape(l,h,p,n) if and only if

$$l>n,\ \ h\geq p,\ \text{ and }\ \gcd(l,h,p,n)=1. \tag{1.1}$$

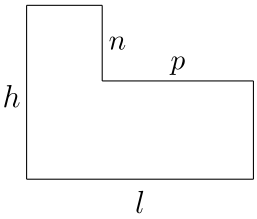
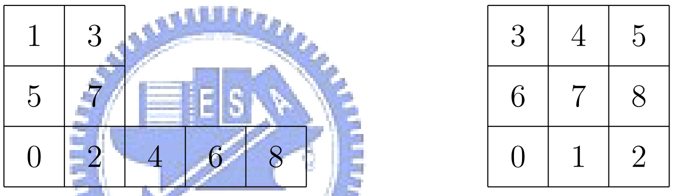


Figure 1: An L-shape with parameters.



$DL(9;2,5)$ $DL(9;1,6)$

Figure 2: Two examples of L-shapes.

The *diameter* $d(N;s_1,s_2)$ of a double-loop network $DL(N;s_1,s_2)$ is the largest distance between any pair of nodes. It represents the maximum transmission delay between two nodes. Thus it is desirable to minimize the diameter and this is the problem discussed by many authors; see [2, 8, 10, 11, 13, 17, 19]. Let $d(N)$ denote the best possible diameter of a double-loop network with N nodes. Wong and Coppersmith [19] showed that

$$d(N)\geq\left\lceil\sqrt{3N}\right\rceil-2.$$

Given an N, it is desirable to find a double-loop network $DL(N;s_1,s_2)$ with its diameter being equal to $d(N)$. Since the diameter of a double-loop network

$DL(N;s_1,s_2)$ can be readily computed from the dimensions of its L-shape, one method is to start with a desirable L-shape and then asks whether there exist s_1 and s_2 to realize it. Three algorithms have been proposed for finding the steps of a double-loop network with N nodes: the Smith normalization method [2, 11], the sieve method [6, 15], and the Chan-Chen-Hong's algorithm (the CCH algorithm for short) [5].

While Cheng and Hwang [8] have given an elegant $O(\log N)$-time algorithm to find the L-shape of a double-loop network $DL(N;s_1,s_2)$, it is an open problem whether the steps of a double-loop network can be found in $O(\log N)$ time. Both the Smith normalization method [2, 11] and the CCH algorithm [5] take $O((\log N)^2)$ time; see [5]. The exact time complexity analysis for the sieve method is not known; see also [5].

Wong and Coppersmith [19] proved that $TL(N;s_1,s_2,s_3)$ is strongly connected if and only if $\gcd(N,s_1,s_2,s_3)=1$. The MDD for a triple-loop network is a 3-dimensional array with each step in the x_i-axis signifying an s_i-step. Unfortunately, the MDD for a triple-loop network does not have a uniform nice shape like the L-shape, and this fact has really hampered the study of triple-loop networks. Aguiló, Fiol and Garcia [3] overcame this difficulty by skipping the triple-loop network and going directly to a nice 3-dimensional shape which they called *hyper-L tile*. This hyper-L tile is characterized by three parameters l,m,n, and is highly structured and symmetrical (see Figure 3). Note that l,m,n are integers, $m \geq n \geq 0$, and $l > m+n$. They used the hyper-L tile to derive a dense family of triple-loop networks which has the property

$$N(D) \geq \frac{2}{27}(D+3)^3 \approx 0.074D^3 + O(D^2),$$

where $N(D)$ is the maximum number of nodes in a triple-loop network for a fixed diameter D. Note that when a family of triple-loop networks has a good N-D ratio, we say it is dense.

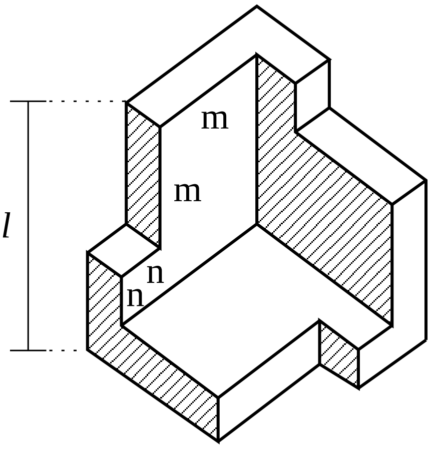


Figure 3: A hyper-L tile.

Also, Aguiló-Gost [1] presented a new type of hyper-L tiles which is characterized by three parameters h, m, n, and is also highly structured and symmetrical (see Figure 4). Aguiló-Gost [1] used it to derive a new dense family of triple-loop networks which has the property

$$N(D) \geq \frac{1485}{27^3}D^3 \approx 0.075D^3 + O(D^2).$$

For convenience, call this hyper-L tile the *hyper-L_1 tile.*

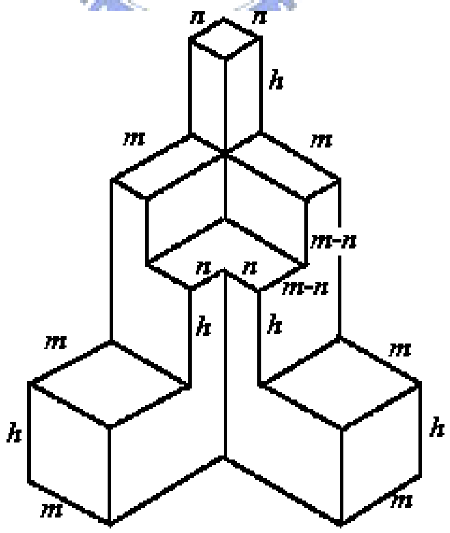


Figure 4: A hyper-L_1 tile.

While the hyper-L and the hyper-L_1 tiles seem to be promising tools for studying the triple-loop network, we must be able to verify that those hyper-L tiles producing good results are indeed the MDDs of some triple-loop networks. In [7], Chen et. al. have presented necessary and sufficient conditions for the existence of hyper-L triple-loop networks; see also [3].

In this thesis, we first prove that there exists a family of double-loop networks such that the sieve method requires $\Omega((\log N)^{3/2})$ time to find the steps for each double-loop network in this family. We then propose a simple $O(\log N)$-time algorithm to find the steps of a double-loop network with N nodes. We also give necessary and sufficient conditions for the existence of hyper-L_1 triple-loop networks.

This thesis is organized as follows: In Section 2, we briefly describe the Smith normalization method, the sieve method, and the CCH algorithm. In Section 3, we propose a simple $O(\log N)$-time algorithm to find the steps of a double-loop network with N nodes. In Section 4, we give necessary and sufficient conditions for the existence of hyper-L_1 triple-loop networks.

2 The Smith normalization method, the sieve method, and the CCH algorithm

For completeness of this thesis, we briefly describe the Smith normalization method, the sieve method, and the CCH algorithm in this section. The sieve method is based on the sieve method in number theory and is very simple and easy to implement. The CCH algorithm is based on the Smith normalization method of Aguiló, Esqué and Fiol [2, 11], but unlike the Smith normalization method, it does not require any matrix operations and thus greatly simplifies the computation of the Smith normalization method.

Given an L-shape, Aguiló and Fiol [2], and also Esqué et al. [11] proposed the following method for computing s_1 and s_2 such that $DL(N; s_1, s_2)$ realizes L.

THE-SMITH-NORMALIZATION-METHOD [2, 11].

Input: l, h, p, n of an L-shape L, where $l > n$, $h \geq p$, and $\gcd(l, h, p, n) = 1$.

Output: s_1 and s_2 such that $DL(N; s_1, s_2)$ realizes the L-shape $L(l, h, p, n)$.

1. Let

$$\mathcal{M} = \begin{pmatrix} l & -p \\ -n & h \end{pmatrix},$$

$\mathcal{M}_0 = \mathcal{M}$, $i = 0$, $j = 0$, $k = 0$.

2. Repeat the sub-steps 2.1-2.2 until the (1,1) element of $\mathcal{M}_j$ divides both the (2,1) element and the (1,2) element of $\mathcal{M}_j$.

2.1 If the (1,1) element of $\mathcal{M}_j$ does not divide the (2,1) element of $\mathcal{M}_j$, then let $i = i + 1$, $j = j + 1$, and find a nonsingular unimodular (i.e., determinant $\pm$ 1) integral matrix $\mathcal{L}_i$ such that the (1,1) element of $\mathcal{M}_j = \mathcal{L}_i \mathcal{M}_{j-1}$ is the greatest common divisor of the first column of $\mathcal{M}_{j-1}$.

2.2 If the (1,1) element of $\mathcal{M}_j$ does not divide the (1,2) element of $\mathcal{M}_j$, then let $j = j + 1$, $k = k + 1$, and find a nonsingular unimodular integral matrix $\mathcal{R}_k$ such that the (1,1) element of $\mathcal{M}_j = \mathcal{M}_{j-1} \mathcal{R}_k$ is the greatest common divisor of the first row of $\mathcal{M}_{j-1}$.

3. If the (2,1) element of $\mathcal{M}_j$ is not zero, then let $i = i + 1$, $j = j + 1$, and find a nonsingular unimodular integral matrix $\mathcal{L}_i$ to make the (2,1) element of $\mathcal{M}_j = \mathcal{L}_i \mathcal{M}_{j-1}$ zero.

4. If the (1,2) element of $\mathcal{M}_j$ is not zero, then let $j = j + 1$, $k = k + 1$, and find a nonsingular unimodular integral matrix $\mathcal{R}_k$ to make the (1,2) element of $\mathcal{M}_j = \mathcal{M}_{j-1} \mathcal{R}_k$ zero.

5. If the (1,1) element of $\mathcal{M}_j$ does not divide the (2,2) element of $\mathcal{M}_j$, then add column 2 of $\mathcal{M}_j$ to column 1 of $\mathcal{M}_j$ and go to Step 2.

6. Now $\mathcal{M}_j$ is the Smith normal form of $\mathcal{M}$, i.e.,

$$\mathcal{M}_j = \mathcal{L}_i \cdots \mathcal{L}_2 \mathcal{L}_1 \mathcal{M} \mathcal{R}_1 \mathcal{R}_2 \cdots \mathcal{R}_k = \mathcal{S}(\mathcal{M}) = \begin{pmatrix} 1 & 0 \\ 0 & N \end{pmatrix}.$$

Let $\mathcal{L} = \mathcal{L}_i \cdots \mathcal{L}_2\mathcal{L}_1$. If

$$\mathcal{L} = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix},$$

then let $s_1 = \gamma \pmod N$ and let $s_2 = \delta \pmod N$. Return s_1, s_2.

Given an L-shape, Chen and Hwang [6] (see also [15]) proposed the following method, which is based on the sieve method in number theory, for computing s_1 and s_2 such that $DL(N; s_1, s_2)$ realizes L.

THE-SIEVE-METHOD [6].

Input: l, h, p, n of an L-shape L, where $l > n$, $h \geq p$, and $\gcd(l, h, p, n) = 1$.

Output: s_1 and s_2 such that $DL(N; s_1, s_2)$ realizes the L-shape $L(l, h, p, n)$.

1. Let $k = 0$ and let $F =$ the set of prime factors of N.

2. Let

$a_k = kn + h,$

$b_k = kl + p,$

$F_k =$ the set of prime factors of $\gcd(a_k, b_k)$.

3. If $f \notin F_k$ for all $f \in F$, then $s_1 = a_k \pmod N$ and $s_2 = b_k \pmod N$ realize L; otherwise, if $f \in F_k$ for any $f \in F$, then go to Step 2.

We now prove that there exists a family of double-loop networks such that the sieve method requires $\Omega((\log N)^{3/2})$ time to find the steps for each double-loop network in this family. The following lemma will be used in the proof.

Lemma 1 *Let $p_1, p_2, \cdots, p_t$ be the smallest t primes, where $t \geq 2$ and $p_1 < p_2 < \cdots < p_t$. If $N = p_1 \times p_2 \times \cdots \times p_t$, then $p_t \geq \sqrt{\log N}$.*

Proof. Suppose $N = p_1 \times p_2 \times \cdots \times p_t$. Then $N \leq (p_t)! \leq p_t^{p_t}$. Therefore $\log N \leq p_t \log p_t$ and $\log\log N \leq \log p_t + \log\log p_t \leq 2\log p_t$. So $\log p_t \geq \frac{1}{2}\log\log N = \log\sqrt{\log N}$ and we have $p_t \geq \sqrt{\log N}$. ∎

Theorem 2 *There exists a family of double-loop networks such that the sieve method requires $\Omega((\log N)^{3/2})$ time to find the steps for each double-loop network in this family.*

Proof. Let t be an integer such that $2 \leq t \leq 100000$. Let $p_1, p_2, \cdots, p_t$ be the smallest t primes and $p_1 < p_2 < \cdots < p_t$. Let

$$d = p_1 \times p_2 \times \cdots \times p_{t-1}.$$

It is not difficult to verify that for each t in $\{2, 3, \cdots, 100000\}$, $p_t \leq 2p_{t-1}$ and thus $\left\lfloor \frac{2d}{p_t} \right\rfloor \geq 1$. Since $2d$ is not divisible by p_t, $\left\lceil \frac{2d}{p_t} \right\rceil$ and $\left\lfloor \frac{2d}{p_t} \right\rfloor$ are two consecutive integers. Therefore

$$\left\lceil \frac{2d}{p_t} \right\rceil - \left\lfloor \frac{2d}{p_t} \right\rfloor = 1 \tag{2.2}$$

and

$$\gcd\left(\left\lceil \frac{2d}{p_t} \right\rceil, \left\lfloor \frac{2d}{p_t} \right\rfloor\right) = 1. \tag{2.3}$$

Let

$$l = p_t \left\lceil \frac{2d}{p_t} \right\rceil - d,\ h = d,\ p = d,\ n = p_t \left\lfloor \frac{2d}{p_t} \right\rfloor - d.$$

We claim that there exists a $DL(N; s_1, s_2)$ realizing the L-shape(l, h, p, n). To prove this claim, we have to show that

$$l > 0,\ h > 0,\ p \geq 0,\ n \geq 0,\ l \geq p,\ h \geq n,\ lh - pn = N,$$

and to show that the three conditions in (1.1) hold. It is clear that $l > p_t\left(\frac{2d}{p_t}\right) - d = d > 0$, $h = d > 0$, $p = d \geq 0$. Since $\left\lfloor \frac{2d}{p_t} \right\rfloor \geq 1$, we have $n > p_t\left(\frac{2d}{p_t} - 1\right) - d \geq 0$. We have $l \geq p$ since

$$l = p_t \left\lceil \frac{2d}{p_t} \right\rceil - d \geq p_t \left(\frac{2d}{p_t}\right) - d = d = p.$$

We have $h \geq n$ since

$$h = d = p_t\left(\frac{2d}{p_t}\right) - d \geq p_t\left\lfloor\frac{2d}{p_t}\right\rfloor - d = n.$$

Let

$$N = p_1 \times p_2 \times \cdots \times p_t.$$

Then

$$\begin{aligned} lh - pn &= \left(p_t\left\lceil\frac{2d}{p_t}\right\rceil - d\right)d - d\left(p_t\left\lfloor\frac{2d}{p_t}\right\rfloor - d\right) \\ &= dp_t\left(\left\lceil\frac{2d}{p_t}\right\rceil - \left\lfloor\frac{2d}{p_t}\right\rfloor\right) \\ &= dp_t \quad \text{(by (2.2))} \\ &= N. \end{aligned}$$

By (2.2), we have $l > n$. Since $h = p$, we have $h \geq p$. Note that

$$\gcd(p_t, d) = \gcd(p_t, p_1 \times p_2 \times \cdots \times p_{t-1}) = 1. \tag{2.4}$$

Thus

$$\begin{aligned} \gcd\left(l, h, p, n\right) &= \gcd\left(p_t\left\lceil\frac{2d}{p_t}\right\rceil - d, d, d, p_t\left\lfloor\frac{2d}{p_t}\right\rfloor - d\right) \\ &= \gcd\left(p_t\left\lceil\frac{2d}{p_t}\right\rceil, p_t\left\lfloor\frac{2d}{p_t}\right\rfloor, d\right) \\ &= \gcd\left(\left\lceil\frac{2d}{p_t}\right\rceil, \left\lfloor\frac{2d}{p_t}\right\rfloor, d\right) \quad \text{(by (2.4))} \\ &= 1. \quad \text{(by (2.3))} \end{aligned}$$

We now claim that for each t in $\{2, 3, \cdots, 100000\}$, the sieve method requires $\Omega((\log N)^{3/2})$ time to find the steps for each double-loop network with L-shape L(l, h, p, n). Since $N = p_1 \times p_2 \times \cdots \times p_t$, in the sieve method we will have

$$F = \{p_1, p_2, \cdots, p_t\}.$$

Since $\gcd(a_0, b_0) = \gcd(h, p) = d$, we have

$$F_0 = \{p_1, p_2, \ldots, p_{t-1}\}.$$

Since $\gcd(a_1, b_1) = \gcd(n+h, l+p) = \gcd(p_t \left\lfloor \frac{2d}{p_t} \right\rfloor - d + d, p_t \left\lceil \frac{2d}{p_t} \right\rceil - d + d) = \gcd(p_t \left\lfloor \frac{2d}{p_t} \right\rfloor, p_t \left\lceil \frac{2d}{p_t} \right\rceil)$, by (2.3) we have $\gcd(a_1, b_1) = p_t$ and therefore

$$F_1 = \{p_t\}.$$

Recall that $p_1, p_2, \cdots, p_t$ are the smallest t primes and $p_1 < p_2 < \cdots < p_t$; i.e., $p_1 = 2$, $p_2 = 3$, $p_3 = 5$, $\cdots$. Note that if $f \in F$ appears in F_k for some k and k_f is the smallest such k, then f appears in every f^{th} k after k_f. Therefore

$$p_1 \in F \text{ appears in } \gcd(a_0, b_0),\ \gcd(a_2, b_2),\ \gcd(a_4, b_4),\ \gcd(a_6, b_6),\ \text{etc},$$

$$p_2 \in F \text{ appears in } \gcd(a_0, b_0),\ \gcd(a_3, b_3),\ \gcd(a_6, b_6),\ \gcd(a_9, b_9),\ \text{etc},$$

$$p_3 \in F \text{ appears in } \gcd(a_0, b_0),\ \gcd(a_5, b_5),\ \gcd(a_{10}, b_{10}),\ \gcd(a_{15}, b_{15}),\ \text{etc},$$

$$p_4 \in F \text{ appears in } \gcd(a_0, b_0),\ \gcd(a_7, b_7),\ \gcd(a_{14}, b_{14}),\ \gcd(a_{21}, b_{21}),\ \text{etc},$$

$$\cdots$$

$$p_{t-1} \in F \text{ appears in } \gcd(a_0, b_0),\ \gcd(a_{p_{t-1}}, b_{p_{t-1}}),\ \gcd(a_{p_{t-1}\times 2}, b_{p_{t-1}\times 2}),\ \text{etc},$$

$$p_t \in F \text{ appears in } \gcd(a_1, b_1),\ \gcd(a_{p_t+1}, b_{p_t+1}),\ \gcd(a_{p_t\times 2+1}, b_{p_t\times 2+1}),,\ \text{etc}.$$

Thus the first k such that $f \notin F_k$ for all $f \in F$ is p_t. By Lemma 1, $p_t \geq \sqrt{\log N}$. Since each iteration of the sieve method involves the Euclidean algorithm, the sieve method requires $\Omega((\log N)^{3/2})$ time and we have this theorem. ∎

We now describe the CCH algorithm. Given an L-shape L, Chan, Chen and Hong [5] proposed the following algorithm for computing s_1 and s_2 such that $DL(N; s_1, s_2)$ realizes L. For completeness, we append the proof of the correctness of the CCH algorithm in the appendix.

The CCH algorithm [5].

Input: l, h, p, n of an L-shape L, where $l > n$, $h \geq p$, and $\gcd(l, h, p, n) = 1$.

Output: s_1 and s_2 such that $DL(N; s_1, s_2)$ realizes L.

1. Find $r_1 = \gcd(l, -n)$.

2. Find integers α_1 and β_1 such that $\alpha_1 l + \beta_1(-n) = r_1$.

3. Find $r_2 = \gcd(r_1, -\alpha_1 p + \beta_1 h)$.

4. Find integers α_2 and β_2 such that $\alpha_2 r_1 + \beta_2(-\alpha_1 p + \beta_1 h) = r_2$ and $\gcd(\beta_2, r_2) = 1$.

5. $s_1 = \alpha_2 n - \beta_2 h \pmod N$ and $s_2 = \alpha_2 l - \beta_2 p \pmod N$.

In [5], Step 4 is performed by the following algorithm.

ALGORITHM-MODIFIED-EUCLIDEAN [5].

Input: Integers a and b, not both zero, and $r = \gcd(a, b)$.

Output: Integers x and y such that $xa + yb = r$ and $\gcd(y, r) = 1$.

```
1. Find integers $\alpha$ and $\beta$ such that $\alpha a + \beta b = r$.
2. If $\gcd(\beta, r) = 1$, then let $x = \alpha$, $y = \beta$, return $x, y$ and stop this algorithm.
3. Let $k = \gcd(\beta, r)$, $r' = r$, and $d = k$.
4. WHILE $(d > 1)$ DO
     BEGIN
     $r' = r'/d$;
     $d = \gcd(r', k)$;
     END
5. Let $a' = a/r$, $b' = b/r$, $x = \alpha + r'b'$ and $y = \beta - r'a'$. Return $x$, $y$.
```

For example, let $l = 5$, $h = 3$, $p = 3$, and $n = 2$. Then the CCH algorithm derives

$$r_1 = 1,\ \alpha_1 = 1,\ \beta_1 = 2,\ r_2 = 1,\ \alpha_2 = -2,\ \text{and } \beta_2 = 1.$$

Thus $N = 9$,

$$s_1 = -7 \pmod 9 = 2, \text{ and } s_2 = -13 \pmod 9 = 5.$$

It can be verified from Figure 2 that $DL(9; 2, 5)$ realizes L-shape(5,3,3,2).

3 Our algorithm

Our algorithm is based on the CCH algorithm and therefore unlike the Smith normalization method, our algorithm does not require any matrix operations.

It is well-known that

Lemma 3 *If a and b are integers, not both zero, then there exist integers α and β such that $\alpha a + \beta b = \gcd(a, b)$.*

It is known that $\gcd(a, b) = \gcd(|a|, |b|)$ and if $|b| \geq |a| > 0$, then α, β, and $\gcd(a, b)$ can be found in $O(\log |a|)$ time by using the Euclidean algorithm [9].

Chan et. al. [5] proved that

Lemma 4 [5] *If α, a, β, b are integers, not all zero, such that $\alpha a + \beta b = 1$, then $\gcd(a, \beta) = 1$.*

Step 4 of the CCH algorithm is based on Theorem 5 described below.

Theorem 5 [5] *If a and b are integers, not both zero, then there exist integers x and y such that $xa + yb = \gcd(a, b)$ and $\gcd(y, \gcd(a, b)) = 1$.*

Recall that $N = lh - pn$. It is obvious that Steps 1, 2, and 3 of the CCH algorithm can be done in $O(\log N)$ time by using the Euclidean algorithm. Step 5 of the CCH algorithm takes $O(1)$ time. Since Step 4 of the CCH algorithm takes $O((\log N)^2)$ time (see [5] for details), the CCH algorithm takes $O((\log N)^2)$ time.

We thus conclude that if Step 4 of the CCH algorithm can be done in $O(\log N)$ time, then the CCH algorithm takes only $O(\log N)$ time and the steps of a double-loop network with N nodes can be found in $O(\log N)$ time. The key observation of our algorithm is that Theorem 5 can be proved in another way and this new proof leads to an $O(\log N)$-time implementation for Step 4 of the CCH algorithm.

A new proof for Theorem 5. Set $r = \gcd(a, b)$ for easy writing. By Lemma 3, there exist integers α and β such that

$$\alpha a + \beta b = r.$$

If $\gcd(\beta, r) = 1$, then we are done. In the following, assume that $\gcd(\beta, r) = k > 1$. Let r' be the largest integer such that

$$r' \mid r \text{ and } \gcd(r', k) = 1. \tag{3.5}$$

Then either $r' = 1$ or $r' > 1$. In the former case, every prime factor of r is also a prime factor of k. In the latter case, every prime factor of r is either a prime factor of k or a prime factor of r'.

Let

$$a' = a/r, \text{ and } b' = b/r.$$

Note that $\gcd(r', \beta) = 1$; otherwise, we will have $\gcd(\beta, r) > k$. Since $\alpha a + \beta b = r$, we have

$$\alpha a' + \beta b' = 1.$$

By Lemma 4, we have $\gcd(a', \beta) = 1$. Since $\gcd(a', \beta) = 1$ and $k \mid \beta$, we have $\gcd(a', k) = 1$. Since $k \mid \beta$ and $\gcd(r', k) = 1$ and $\gcd(a', k) = 1$, we have

$$\gcd(\beta - r'a', k) = 1. \tag{3.6}$$

Since $\gcd(r', \beta) = 1$ and $r' \mid r'a'$, we have

$$\gcd(\beta - r'a', r') = 1. \tag{3.7}$$

Recall that $r' = 1$ or $r' > 1$. In the former case, by (3.6), by (3.7), and by the fact that every prime factor of r is also a prime factor of k, we have

$$\gcd(\beta - r'a', r) = 1.$$

In the latter case, by (3.6), by (3.7), and by the fact that every prime factor of r is either a prime factor of k or a prime factor of r', we also have

$$\gcd(\beta - r'a', r) = 1.$$

Let

$$x = \alpha + r'b' \text{ and } y = \beta - r'a'. \tag{3.8}$$

Then

$$xa + yb = (\alpha + r'b')a + (\beta - r'a')b = r$$

and

$$\gcd(y, r) = \gcd(\beta - r'a', r) = 1.$$

We proved the theorem. ∎

The following lemma provides an efficient way to find r' in (3.8), which is the largest integer satisfying (3.5).

Lemma 6 *Let r and k be positive integers such that $k \mid r$ and $k > 1$. Then*

$$r' = \frac{r}{\gcd(k^{\lfloor \log_2 r \rfloor}, r)}$$

is the largest integer satisfying (3.5).

Proof. Assume that

$$k = p_1^{s_1} p_2^{s_2} \cdots p_m^{s_m},$$

where $p_i's$ are distinct prime factors of k. Also assume that

$$r = p_1^{t_1} p_2^{t_2} \cdots p_m^{t_m} p_{m+1}^{t_{m+1}} p_{m+2}^{t_{m+2}} \cdots p_n^{t_n},$$

where $p_j's$ are distinct prime factors of r. Note that when $p_{m+1}, p_{m+2}, \cdots, p_n$ do not exist (this case occurs when k contains every prime factor of r), we will simple say that $p_{m+1}^{t_{m+1}} p_{m+2}^{t_{m+2}} \cdots p_n^{t_n} = 1$. It is clear that the largest integer satisfying (3.5) is

$$r' = p_{m+1}^{t_{m+1}} p_{m+2}^{t_{m+2}} \cdots p_n^{t_n}$$

and it suffices to prove that

$$\frac{r}{\gcd(k^{\lfloor \log_2 r \rfloor}, r)} = p_{m+1}^{t_{m+1}} p_{m+2}^{t_{m+2}} \cdots p_n^{t_n}.$$

Note that

$$k^{\lfloor \log_2 r \rfloor} = p_1^{\lfloor \log_2 r \rfloor s_1} p_2^{\lfloor \log_2 r \rfloor s_2} \cdots p_m^{\lfloor \log_2 r \rfloor s_m}.$$

Since 2 is the smallest prime, we have

$$t_i \leq \lfloor \log_2 r \rfloor, \text{ for all } i, \ 1 \leq i \leq n.$$

Therefore

$$t_i \leq \lfloor \log_2 r \rfloor s_i, \text{ for all } i, \ 1 \leq i \leq m.$$

Thus

$$\gcd(k^{\lfloor \log_2 r \rfloor}, r) = p_1^{t_1} p_2^{t_2} \cdots p_m^{t_m}.$$

So

$$\frac{r}{\gcd(k^{\lfloor \log_2 r \rfloor}, r)} = \frac{p_1^{t_1} p_2^{t_2} \cdots p_m^{t_m} p_{m+1}^{t_{m+1}} p_{m+2}^{t_{m+2}} \cdots p_n^{t_n}}{p_1^{t_1} p_2^{t_2} \cdots p_m^{t_m}} = p_{m+1}^{t_{m+1}} p_{m+2}^{t_{m+2}} \cdots p_n^{t_n}.$$

∎

The new proof of Theorem 5 and Lemma 6 lead to the following new algorithm for finding x and y in Theorem 5.

ALGORITHM-NEW-MODIFIED-EUCLIDEAN.

Input: Integers a and b, not both zero, and $r = \gcd(a, b)$.

Output: Integers x and y such that $xa + yb = r$ and $\gcd(y, r) = 1$.

1. Find integers α and β such that $\alpha a + \beta b = r$.

2. Find $k = \gcd(\beta, r)$.

3. If $k = 1$, then let $x = \alpha$, $y = \beta$, return x, y and stop this algorithm.

4. If $k \geq 2$, find

$$r' = \frac{r}{\gcd(k^{\lfloor \log_2 r \rfloor}, r)}.$$

5. Let $a' = a/r$, $b' = b/r$, $x = \alpha + r'b'$ and $y = \beta - r'a'$. Return x, y.

The correctness of ALGORITHM-NEW-MODIFIED-EUCLIDEAN follows from Lemma 6. We now prove that:

Theorem 7 *If we use ALGORITHM-NEW-MODIFIED-EUCLIDEAN instead of ALGO RITHM-MODIFIED-EUCLIDEAN in Step 4 of the CCH algorithm, then the CCH algorithm takes $O(\log N)$ time to find the steps of a double-loop network.*

Proof. Recall that $N = lh - pn$. Since Steps 1, 2, 3, and 5 of the CCH algorithm take $O(\log N)$ time, it suffices to prove that ALGORITHM-NEW-MODIFIED-EUCLIDEAN takes $O(\log N)$ time. When Step 4 of the CCH algorithm is performed, inputs to ALGORITHM-NEW-MODIFIED-EUCLIDEAN are

$$a = r_1,$$
$$b = -\alpha_1 p + \beta_1 h,$$
$$r = \gcd(r_1, -\alpha_1 p + \beta_1 h).$$

In the following, Step i refers to Step i in ALGORITHM-NEW-MODIFIED-EUCLIDEAN.

Since $r_1 = \gcd(l, -n)$, we have

$$r_1 > 0,\ r_1 \leq N,$$

and

$$\min\{|a|, |b|\} = \min\{|r_1|, |-\alpha_1 p + \beta_1 h|\} \leq |r_1| = r_1 \leq N.$$

Therefore Step 1 can be done in $O(\log N)$ time by using the Euclidean algorithm. Since $r = \gcd(r_1, -\alpha_1 p + \beta_1 h)$, we have

$$r > 0,\ r \leq r_1 \leq N,$$

and

$$\min\{|\beta|, |r|\} \leq |r| = r \leq N.$$

Therefore, in Step 2, finding $k = \gcd(\beta, r)$ can be done in $O(\log N)$ time by using the Euclidean algorithm. It is obvious that Step 3 and Step 5 can be done in $O(1)$ time. It remains to prove that in Step 4, finding $r' = \frac{r}{\gcd(k^{\lfloor \log_2 r \rfloor}, r)}$ can also be done in $O(\log N)$ time. Note that computing $k^{\lfloor \log_2 r \rfloor}$ takes $O(\log \lfloor \log_2 r \rfloor) = O(\log \log N)$ time. Since $r > 0$ and $r \leq N$, we have

$$\min\{|k^{\lfloor \log_2 r \rfloor}|, |r|\} \leq |r| = r \leq N.$$

Therefore, finding $\gcd(k^{\lfloor \log_2 r \rfloor}, r)$ takes $O(\log r) = O(\log N)$ time. Hence finding $r' = \frac{r}{\gcd(k^{\lfloor \log_2 r \rfloor}, r)}$ takes a total

$$O(\log \log N) + O(\log N) + 1 = O(\log N)$$

time, where +1 is for the division. From the above, Step 4 can be done in $O(\log N)$ time. We have this theorem. ∎

4 Necessary and sufficient conditions for the existence of hyper-L_1 triple-loop networks

The following two lemmas will be used in the remaining discussions.

Lemma 8 *If a, m, b, n are integers, not all zero, such that $am - bn = 1$, then* $\gcd(a, n) = 1$.

Proof. Assume that $am - bn = 1$ and $\gcd(a, n) = k$. Then $k \mid a$ and $k \mid n$. Thus $k \mid am - bn = 1$. So $k = 1$. ∎

Lemma 9 *If m and n are integers, not both zero, and* $\gcd(m, n) = 1$, *then there exist integers a and b such that $am - bn = 1$ and* $\gcd(a, 2m + n) = 1$.

Proof. By Lemma 3, there exist integers a and b such that $am - bn = 1$. By Lemma 8, we have

$$\gcd(a, n) = 1. \tag{4.9}$$

If $\gcd(a, 2m+n) = 1$, then we are done. In the following, assume that $\gcd(a, 2m+n) = d > 1$. Let

$$a = pd$$

and

$$2m + n = qd.$$

Then $\gcd(p, q) = 1$. Since $\gcd(m, n) = 1$, we have

$$\gcd(2m+n, n) = \gcd(2m, n) = \begin{cases} 1 & \text{if } n \text{ is odd,} \\ 2 & \text{if } n \text{ is even.} \end{cases}$$

If $\gcd(2m+n, n) = 1$, then clearly $\gcd(qd, n) = 1$ and thus $\gcd(d, n) = 1$. Now suppose that $\gcd(2m+n, n) = 2$. Then $\gcd(qd, n) = 2$; therefore $\gcd(d, n) = 1$ or $\gcd(d, n) = 2$. If $\gcd(d, n) = 2$, then $2 \mid a$ and we have $\gcd(a, n) \geq 2$; this contradicts with(4.9). From the above, we have

$$\gcd(d, n) = 1. \tag{4.10}$$

Let $q = st$, where s is the largest factor of q such that

$$\gcd(s, d) = 1. \tag{4.11}$$

That is, s (t) contains those prime factors of q that are relative prime (not relative prime) to d. (As an example, if $q = 2^2 \cdot 3^2 \cdot 7$ and $d = 2 \cdot 3^2$, then $s = 7$ and $t = 2^2 \cdot 3^2$.) Then

$$\gcd(s, t) = 1. \tag{4.12}$$

Since $\gcd(p, q) = 1$ and $q = st$, we have

$$\gcd(p, s) = 1. \tag{4.13}$$

Since t contains those prime factors of q that are not relative prime to d, by (4.10), we have

$$\gcd(t, n) = 1. \tag{4.14}$$

Let $a' = a + sn$ and $b' = b + sm$. Then

$$a'm - b'n = (a + sn)m - (a + sm)n = am - bn = 1.$$

Moreover,

$$\begin{aligned} & \gcd(a', 2m + n) \\ = \ & \gcd(a + sn, 2m + m) \\ = \ & \gcd(pd + sn, qd) \\ = \ & \gcd(pd + sn, q) \text{ (by (4.10) and (4.11))} \\ = \ & \gcd(pd + sn, st) \\ = \ & \gcd(pd + sn, s) \text{ (by (4.12) and (4.14))} \\ = \ & 1 \text{ (by (4.11) and (4.13)).} \end{aligned}$$

Thus we have this lemma. ■

Let $\text{HL}_1(h, m, n)$ denote a hyper-L_1 tile with parameters h, m, n. Aguiló-Gost [1] defined

$$M_1(h, m, n) = \begin{pmatrix} n & -m & -m \\ n & n + m & -m \\ 2h & h & 2h - n \end{pmatrix}$$

and derived that the diameter of $\text{HL}_1(h, m, n)$ is given by

$$D(h, m, n) = \max\{3m + h + n, 2m + 2h + n, 3h + 3n\} - 3. \tag{4.15}$$

Note that two sides labelled length n in Figure 5 in [1] are actually of length $m - n$; see Figure 5 This flaw can be verified by checking the lengths of the sides of the topmost $n \times n$ square and the lengths of the sides of the rightmost $m \times h$ rectangle. See Figure 4 for a correction of Figure 5.

Thus matrix M_1 should be

$$M_1(h,m,n) = \begin{pmatrix} n & -m & -m \\ n & n+m & -m \\ 2h & h & h+m-n \end{pmatrix}$$

and the diameter of $\mathrm{HL}_1(h,m,n)$ should be

$$D(h,m,n) = \max\{3m+h+n, 2m+2h+n, \\ 3h+3n, m+2h+3n\} - 3. \tag{4.16}$$

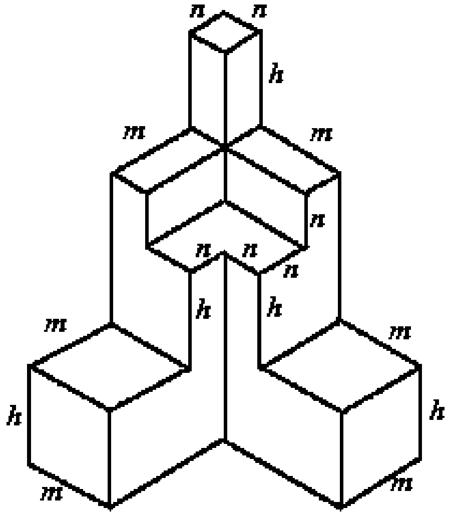


Figure 5: The Fig. 5 in [1].

Note that the difference between the diameters derived by (4.15) and by (4.16) can be quite large. To see this, let

$$\begin{aligned} h &= 2t-1-k \\ m &= 2t-1 \\ n &= t+k, \end{aligned}$$

where t and k are positive integers chosen in such a way that both $\gcd(m,n) = 1$ and $3 \nmid m-n$ are satisfied. Then the diameter derived by (4.15) is $9t-6$, while the diameter derived by (4.16) is $9t-6+k$. The difference between the two diameters is k. As an example, when $k=5$, we can choose $t=10$, $h=14$, $m=19$, and $n=15$.

Aguiló-Gost [1] observed that $\mathrm{HL}_1(h,m,n)$ tessellates the space. By studying the distribution of node 0 in the space, Aguiló-Gost obtained

$$M^T \times \begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \pmod{N} \quad \text{or}$$

$$M^T \times \begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} N \text{ for some integers } \alpha, \beta, \gamma. \tag{4.17}$$

Also, $N = \det M$.

For convenience, we call a triple-loop network whose MDD is $\mathrm{HL}_1(h,m,n)$ an $\mathrm{HL}_1(h,m,n)$ triple-loop. We now give a necessary and sufficient condition for the existence of an $\mathrm{HL}_1(h,m,n)$ triple-loop.

Theorem 10 *A necessary and sufficient condition for the existence of an $HL_1(h,m,n)$ triple-loop is* $\gcd(m,n)=1$ *and* $3 \nmid m-n$.

Proof. Note that $N = \det M$. Suppose an $\mathrm{HL}_1(h,m,n)$ triple-loop exists. From (4.17), we have

$$\begin{aligned} &\begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix} \\ &= \left(M^T\right)^{-1} \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} N \\ &= \begin{pmatrix} h(2m+n)+(m-n)(m+n) & -(h(2m+n)+n(m-n)) & -h(2m+n) \\ m(m-n) & h(2m+n)+n(m-n) & -h(2m+n) \\ m(2m+n) & 0 & n(2m+n) \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix}. \end{aligned}$$

Suppose that $\gcd(m,n)=1$ and $3 \nmid m-n$. Since $\gcd(m,n)=1$, by Lemma 9, there exist integers a and b such that $am-bn=1$ and $\gcd(a,2m+n)=1$. Since $\gcd(a,2m+n)=1$, we have $a \neq 0$. Since $\gcd(m,n)=1$,

$$\gcd(m-n,m)=1.$$

Since $\gcd(m,n)=1$, $3\nmid m-n$, and $\gcd(m-n,m)=1$,

$$\gcd(m-n,2m+n)=\gcd(m-n,3m)=\gcd(m-n,3)=1. \tag{4.18}$$

Setting $(\alpha,\beta,\gamma)=(a,0,-b)$, we obtain the solution

$$\begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} h(a+b)(2m+n)+a(m-n)(m+n) \pmod N \\ bh(2m+n)+am(m-n) \pmod N \\ 2m+n \end{pmatrix}.$$

Since $N=\det M$, we have

$$N=(2m+n)(h(2m+n)+n(m-n)). \tag{4.19}$$

Let

$$\phi(a)=\begin{cases} -1 & \text{if } a>0, \\ 1 & \text{if } a<0. \end{cases}$$

From (4.19), $2m+n \mid N$. Since $2m+n \mid N$, there exists an integer k_1 such that

$$\begin{aligned} & h(a+b)(2m+n)+a(m-n)(m+n) \pmod N \\ = \; & k_1(2m+n)+\phi(a)a(m-n)(m+n) \end{aligned}$$

and $0<k_1(2m+n)+\phi(a)a(m-n)(m+n)<N$. Also, there exists an integer k_2 such that

$$\begin{aligned} & bh(2m+n)+am(m-n) \pmod N \\ = \; & k_2(2m+n)+\phi(a)am(m-n) \end{aligned}$$

and $0<k_2(2m+n)+\phi(a)am(m-n)<N$. Therefore

$$\begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} k_1(2m+n)+\phi(a)a(m-n)(m+n) \\ k_2(2m+n)+\phi(a)am(m-n) \\ 2m+n \end{pmatrix}.$$

Note that

$$\begin{aligned}
& \gcd(k_1(2m+n)+\phi(a)a(m-n)(m+n), k_2(2m+n)+\phi(a)am(m-n), 2m+n) \\
= \; & \gcd(a(m-n)(m+n), am(m-n), 2m+n) \\
= \; & \gcd(an(m-n), am(m-n), 2m+n) \\
= \; & \gcd(n(m-n), m(m-n), 2m+n) \quad (\text{by the fact that } \gcd(a, 2m+n)=1) \\
= \; & \gcd(n, m, 2m+n) \quad (\text{by } (4.18)) \\
= \; & \gcd(m, n) \\
= \; & 1.
\end{aligned}$$

So if $\gcd(m,n)=1$ and $3 \nmid m-n$, then clearly

$$\gcd(N, s_1, s_2, s_3) = \gcd(s_1, s_2, s_3) = 1$$

and $TL(N; s_1, s_2, s_3)$ exists.

On the other hand, suppose

$$\gcd(m,n) = d > 1 \text{ or } 3 \mid m-n.$$

In the former case, each s_i, $i=1,2,3$, is a linear combination of terms divisible by d. Furthermore, from (4.19), N is also a linear combination of terms divisible by d. Hence

$$\gcd(N, s_1, s_2, s_3) \geq d > 1$$

and $TL(N; s_1, s_2, s_3)$ does not exist. In the latter case, since $3 \mid m-n$, we have

$$\gcd(2m+n, m-n) = \gcd(3m, m-n) = r \geq 3.$$

Therefore each s_i, $i=1,2,3$, is a linear combination of terms divisible by r. Furthermore, from (4.19), N is also a linear combination of terms divisible by r. Hence

$$\gcd(N, s_1, s_2, s_3) \geq r > 1$$

and $TL(N; s_1, s_2, s_3)$ does not exist. ∎

References

[1] F. Aguiló-Gost, New dense families of triple loop networks, *Disc. Math.* 197/198 (1999) 15-27.

[2] F. Aguiló and M. A. Fiol, An efficient algorithm to find optimal double loop networks, *Disc. Math.* 138 (1995), 15-29.

[3] F. Aguiló, M. A. Fiol and C. Garcia, Triple-loop networks with small transmission delay, *Disc. Math.* 167/168 (1997) 3-16.

[4] J.-C. Bermond, F. Comellas and D. F. Hsu, Distributed loop computer networks: a survey, *J. Parallel Distribut. Comput.* 24 (1995), 2-10.

[5] R. C. Chan, C. Y. Chen and Z. X. Hong, A simple algorithm to find the steps of double-loop networks, *Disc. Appl. Math.* 121 (2002), 61-72.

[6] C. Y. Chen and F. K. Hwang, The minimum distance diagram of double-loop networks, *IEEE Trans. Comput.* 49 (2000), 977-979.

[7] C. Y. Chen, F. K. Hwang, J. S. Lee and S. J. Shih, The existence of hyper-L triple-loop networks, *Disc. Math.* 268 (2003) 287-291.

[8] Y. Cheng and F. K. Hwang, Diameters of weighted double loop networks, *J. Algorithms* 9 (1988), 401-410.

[9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd Ed. (2001), The MIT Press, 856-862.

[10] P. Erdös and D. F. Hsu, Distributed loop networks with minimum transmission delay, *Theoret. Comput. Sci.* 100 (1992), 223-241.

[11] P. Esqué, F. Aguiló, and M. A. Fiol, Double commutative-step diagraphs with minimum diameters, *Disc. Math.* 114 (1993), 147-157.

[12] M. A. Fiol, M. Valero, J. L. A. Yebra, I. Alegre, and T. Lang, Optimization of double-loop structures for local networks, in *Proc. XIX Int. Symp. MIMI'82*, Paris, France (1982), 37-41.

[13] M. A. Fiol, J. L. A. Yebra, I. Alegre, and M. Valero, A discrete optimization problem in local networks and data alignment, *IEEE Trans. Comput.* C-36 (1987), 702-713.

[14] F. K. Hwang, A survey on double-loop networks, *Reliability of Computer and Communication Networks*, Eds: F. Roberts, F. K. Hwang and C. Monma, AMS series (1991), 143-151.

[15] F. K. Hwang, A complementary survey on double-loop networks, *Theoret. Comput. Sci. A* 263 (2001), 211-229.

[16] F. K. Hwang, A survey on multi-loop networks, *Theoret. Comput. Sci. A* 299 (2003), 107-121.

[17] F. K. Hwang and Y. H. Xu, Double loop networks with minimum delay, *Disc. Math.* 66 (1987), 109-118.

[18] J. M. Peha and F. A. Tobagi, Analyzing the fault tolerance of double-loop networks, *IEEE Trans. Network.* 2 (1994), 363-373.

[19] C. K. Wong and D. Coppersmith, A combinatorial problem related to multimodule memory organizations, *J. Assoc. Comput. Mach.* 21 (1974), 392-402.

Appendix

Theorem 11 [5] *The CCH algorithm is correct and it takes at most $O((\log N)^2)$ time.*

Proof. Note that $N = lh - pn$. Let

$$\mathcal{M} = \begin{pmatrix} l & -p \\ -n & h \end{pmatrix}$$

Consider column 1 of $\mathcal{M}$: it contains l and $-n$. After Step 1 is performed, we have $r_1 = \gcd(l, -n)$ and $\alpha_1 l + \beta_1(-n) = r_1$. Let

$$\mathcal{L}_1 = \begin{pmatrix} \alpha_1 & \beta_1 \\ \frac{n}{r_1} & \frac{l}{r_1} \end{pmatrix}.$$

and let $\mathcal{M}_1 = \mathcal{L}_1\mathcal{M}$. Then

$$\mathcal{M}_1 = \begin{pmatrix} \alpha_1 & \beta_1 \\ \frac{n}{r_1} & \frac{l}{r_1} \end{pmatrix}\begin{pmatrix} l & -p \\ -n & h \end{pmatrix} = \begin{pmatrix} r_1 & -\alpha_1 p + \beta_1 h \\ 0 & \frac{N}{r_1} \end{pmatrix}.$$

Consider row 1 of $\mathcal{M}_1$: it contains r_1 and $-\alpha_1 p + \beta_1 h$. After Step 2 is performed, we have $r_2 = \gcd(r_1, -\alpha_1 p + \beta_1 h)$, $\alpha_2 r_1 + \beta_2(-\alpha_1 p + \beta_1 h) = r_2$, and $\gcd(\beta_2, r_2) = 1$. Let

$$\mathcal{R}_1 = \begin{pmatrix} \alpha_2 & \frac{-(-\alpha_1 p + \beta_1 h)}{r_2} \\ \beta_2 & \frac{r_1}{r_2} \end{pmatrix}.$$

and let $\mathcal{M}_2 = \mathcal{M}_1\mathcal{R}_1$. Then

$$\mathcal{M}_2 = \begin{pmatrix} r_1 & -\alpha_1 p + \beta_1 h \\ 0 & \frac{N}{r_1} \end{pmatrix}\begin{pmatrix} \alpha_2 & \frac{-(-\alpha_1 p + \beta_1 h)}{r_2} \\ \beta_2 & \frac{r_1}{r_2} \end{pmatrix} = \begin{pmatrix} r_2 & 0 \\ \frac{N\beta_2}{r_1} & \frac{N}{r_2} \end{pmatrix}.$$

Consider column 1 of $\mathcal{M}_2$: it contains r_2 and $\frac{N\beta 2}{r_1}$. Let $r_3 = \gcd(r_2, \frac{N\beta 2}{r_1})$. Note that in Step 2 we choose $\gcd(\beta_2, r_2) = 1$. Thus

$$r_3 = \gcd(r_2, \tfrac{N\beta 2}{r_1}) = \gcd(r_2, \tfrac{N}{r_1}) = \gcd(r_1, -\alpha_1 p + \beta_1 h, \tfrac{N}{r_1}).$$

We claim that $r_3 = 1$. Suppose this is not true and $r_3 > 1$. Then every entry of $\mathcal{M}_1$ is a multiple of r_3. Since $\mathcal{M}_1 = \mathcal{L}_1\mathcal{M}$, we have

$$\mathcal{M} = \mathcal{L}_1^{-1}\mathcal{M}_1 = \frac{1}{\det(\mathcal{L}_1)}\begin{pmatrix} \frac{l}{r_1} & -\beta_1 \\ -\frac{n}{r_1} & \alpha_1 \end{pmatrix}\begin{pmatrix} r_1 & -\alpha_1 p + \beta_1 h \\ 0 & \frac{N}{r_1} \end{pmatrix}.$$

That is,

$$\mathcal{M} = \frac{1}{\det(\mathcal{L}_1)} \begin{pmatrix} \frac{l}{r_1} & -\beta_1 \\ -\frac{n}{r_1} & \alpha_1 \end{pmatrix} r_3 \begin{pmatrix} \frac{r_1}{r_3} & \frac{-\alpha_1 p+\beta_1 h}{r_3} \\ 0 & \frac{N}{r_1 r_3} \end{pmatrix}.$$

Since $r_3 = \gcd(r_1, -\alpha_1 p + \beta_1 h, \frac{N}{r_1})$,

$$\begin{pmatrix} \frac{r_1}{r_3} & \frac{-\alpha_1 p+\beta_1 h}{r_3} \\ 0 & \frac{N}{r_1 r_3} \end{pmatrix}$$

is integral. Since $\det(\mathcal{L}_1) = \pm 1$, every entry of $\mathcal{M}$ must be a multiple of r_3. Then $\gcd(l, h, p, n) \geq r_3 > 1$; this contradicts with the assumption that $\gcd(l, h, p, n) = 1$. Therefore $r_3 = 1$.

Since $r_3 = \gcd(r_2, \frac{N\beta 2}{r_1})$ and $r_3 = 1$, by Lemma 3, there exist integers α_3 and β_3 such that $\alpha_3 r_2 + \beta_3(\frac{N\beta_2}{r_1}) = 1$. Let

$$\mathcal{L}_2 = \begin{pmatrix} \alpha_3 & \beta_3 \\ \frac{-N\beta_2}{r_1} & r_2 \end{pmatrix}.$$

and let $\mathcal{M}_3 = \mathcal{L}_2 \mathcal{M}_2$. Then

$$\mathcal{M}_3 = \begin{pmatrix} \alpha_3 & \beta_3 \\ \frac{-N\beta_2}{r_1} & r_2 \end{pmatrix} \begin{pmatrix} r_2 & 0 \\ \frac{N\beta_2}{r_1} & \frac{N}{r_2} \end{pmatrix} = \begin{pmatrix} 1 & \frac{\beta_3 N}{r_2} \\ 0 & N \end{pmatrix}.$$

Let

$$\mathcal{R}_2 = \begin{pmatrix} 1 & -\frac{\beta_3 N}{r_2} \\ 0 & 1 \end{pmatrix}.$$

and let $\mathcal{M}_4 = \mathcal{M}_3 \mathcal{R}_2$. Then

$$\mathcal{M}_4 = \begin{pmatrix} 1 & \frac{\beta_3 N}{r_2} \\ 0 & N \end{pmatrix} \begin{pmatrix} 1 & -\frac{\beta_3 N}{r_2} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & N \end{pmatrix} = \mathcal{S}(\mathcal{M}).$$

From the above, $\mathcal{L}_2 \mathcal{L}_1 \mathcal{M} \mathcal{R}_1 \mathcal{R}_2 = \mathcal{S}(\mathcal{M})$. Moreover, $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{R}_1$ and $\mathcal{R}_2$ are unimodular integral matrices. Let $\mathcal{L} = \mathcal{L}_1 \mathcal{L}_2$. Then

$$\mathcal{L} = \begin{pmatrix} \alpha_3 & \beta_3 \\ \frac{-N\beta_2}{r_1} & r_2 \end{pmatrix} \begin{pmatrix} \alpha_1 & \beta_1 \\ \frac{n}{r_1} & \frac{l}{r_1} \end{pmatrix} = \begin{pmatrix} \alpha_3\alpha_1 + \frac{\beta_1 n}{r_1} & \alpha_3\beta_1 + \frac{\beta_3 l}{r_1} \\ \frac{-N\beta_2\alpha_1+r_2 n}{r_1} & \frac{-N\beta_2\beta_1+r_2 l}{r_1} \end{pmatrix}.$$

Using the facts that $N = lh - pn$ and $\alpha_1 l + \beta_1(-n) = r_1$ and $\alpha_2 r_1 + \beta_2(-\alpha_1 p + \beta_1 h) = r_2$, we have $\frac{-N\beta_2\alpha_1+r_2 n}{r_1} = \alpha_2 n - \beta_2 h$ and $\frac{-N\beta_2\beta_1+r_2 l}{r_1} = \alpha_2 l - \beta_2 p$. Thus if $S_1 = \alpha_2 n - \beta_2 h \pmod{N}$ and $s_2 = \alpha_2 l - \beta_2 p \pmod{N}$, then $DL(N; s_1, s_2)$ realizes L.

It is clear that Steps 1, 2, and 3 can be done in $O(\log N)$ time by using the Euclidean algorithm. Step 4 can be done in $O((\log N)^2)$ time by using ALGORITHM-MODIFIED-EUCLIDEAN. Step 5 can be done in $O(1)$ time. Thus the CCH algorithm takes at most $O((\log N)^2)$. ■