Implementation of the TSP based on pattern processing with a graphic processing unit

Kouichi Nitta, Shinichi Ohta, and Osamu Matoba

Department of Computer Science and Systems Engineering, Graduate School of Engineering, Kobe University, Address: Rokkodai 1-1, Nada, Kobe 657-8501, Japan;

ABSTRACT

A graphic processing unit (GPU) is much attractive for large scale information processing. Especially, a GPU is considered to be suitable for SIMD processing to image data. We have developed some methods based on SIMD pattern processing and study on implementation of these methods. In this research, design of GPU implementation for the traveling salesman problem (TSP) is reported. Usefulness of GPU implementation is shown by verification.

Keywords: SIMD pattern operation, traveling salesman problem, Effective implementation, image processing, graphic processing unit

1. INTRODUCTION

We have developed some solutions for problems which computational costs exponentially increase in bit length of a target data. These solutions can be applied to a scheme for SIMD (Single Instruction Multiple Data) type two-dimensional (2D) with image compression. Refs.[1] and[2] have presented methods for prime factorization and the travelling salesman problem (TSP) based on the proposed scheme. Especially, the TSP is important because it is one of the NP complete problems. In optical signal processing, some solutions have proposed.[3,4] However, as well as our method, these solutions cannot achieve the TSP with practical computational costs.

We should study on appropriate implementation for our method to estimate and modify it. In this research, a graphic processing processing unit (GPU) is used as a device for implementation. A GPU has some advantaged features to execute large scale information processing. At First, a GPU achieves high performance SIMD process for 2D images because it has inherently been developed as an image processor. Also, design tools for the GPU is recently prepared for general purpose processing. For example, Ref.[5] reports on a computer generated hologram.

Using such a tool our method can be designed and implemented on a GPU. As results of verification, the GPU implementation is verified that it can solve the TSP much faster than a CPU implementation. Also, we study on implementation of a method for the TSP with pattern processing by use of image compression.

Section 2 describes our method for the TSP. Mathematical principle based on matrix-vector multiplication is briefly summarized. Image representations for the binary matrix and the weight vector are explained. A sequence of SIMD processing corresponding with the matrix vector multiplication is shown for implementation. A method to extract the minimum tour is presented. In Sec. 3, we report a GPU implementation of the processing described in Sec. 3. Then, effectiveness of the GPU implementation is confirmed by comparing that with CPU (Central processing unit) implementation.

2. TSP WITH 2D PATTERN PROCESSING

This section shows the method for the TSP with SIMD 2D pattern processing. First, the principle of the method is described in 2.1. This principle has been reported in Ref.[4] In 2.2, we show a procedure for the TSP with the pattern processing. Block diagrams to explain operations are presented in this subsection.

Optics and Photonics for Information Processing III, edited by Khan M. Iftekharuddin, Abdul Ahad Sami Awwal,
Proc. of SPIE Vol. 7442, 744209 · © 2009 SPIE · CCC code: 0277-786X/09/$18 · doi: 10.1117/12.826851

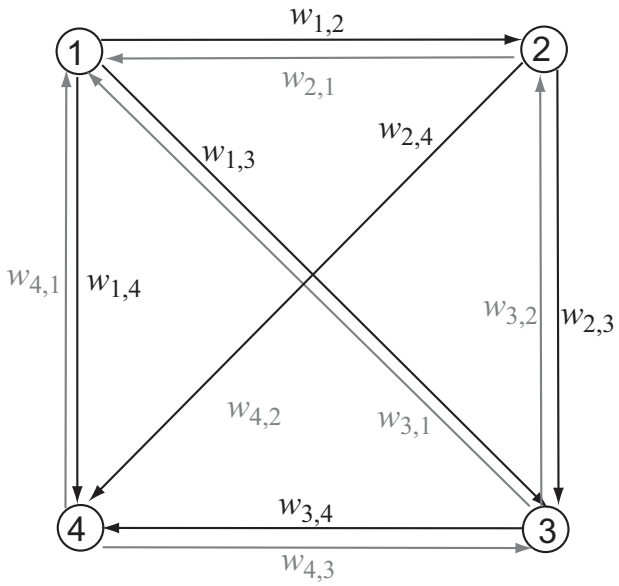

Figure 1. An example of the TSP. This network has four nodes ($N = 4$).

2.1. TSP based on matrix vector multiplication

The TSP is one of the NP complete problems. In the problem, a graph is given and it has a set of pairwise distance. The set is represented as a weight vector in this paper. The purpose of the problem is to derive the shortest tour which passes all nodes once.

A solution for the TSP using 2D SIMD pattern processing is presented in another work.[2] The mathematical procedure of the solution is equivalent to that of an optical solution proposed in Ref.[4] Let us consider a network model described in Fig. 1. From Fig. 1, the number of nodes N is 4. In this research, we treat an asymmetric network. In case that N is n, the number of tour to travel around all nodes is $(n-1)!$. Weights defined as $w_{i,j}$ are given between nodes and the number of the weights is $n(n-1)$. In this situation, length of a tour is given by Eq. (1).

$$l_k = \sum_{j=1}^{n(n-1)} a_{k,j} w_{k,j} \tag{1}$$

In Eq. (1), $a_{k,j}$ =1 if l_k pass the route from the node k to the node j. On the other hand, the value of $a_{k,j}$ is '0' if the route is not used in the tour l_k. $w_{k,j}$ indicates the vaue of weight to transit from the node k to j.

Length of all tours are represented as matrix-vector multiplication. Eq. (2) shows the contents of the

K.N.: E-mail: nitta@kobe-u.ac.jp, FAX: +81(78) 803 6390
O.M.: E-mail: matoba@kobe-u.ac.jp, FAX: +81 (78) 803 6390

operation.

$$
\begin{aligned}
\mathbf{l} = \mathbf{b} \cdot \mathbf{w} &= \begin{pmatrix} b_{0,0} & b_{1,0} & \dots & b_{11,0} \\ b_{0,1} & b_{1,1} & \dots & b_{11,1} \\ b_{0,2} & b_{1,2} & \dots & b_{11,2} \\ b_{0,3} & b_{1,3} & \dots & b_{11,3} \\ b_{0,4} & b_{1,4} & \dots & b_{11,4} \\ b_{0,5} & b_{1,5} & \dots & b_{11,5} \end{pmatrix} \cdot \begin{pmatrix} w_{1,2} \\ w_{1,3} \\ \vdots \\ w_{4,1} \end{pmatrix} \\
&= \begin{pmatrix} b_{0,0}w_{1,2} + b_{1,0}w_{1,3} + \ldots + b_{11,0}w_{4,1} \\ b_{0,1}w_{1,2} + b_{1,1}w_{1,3} + \ldots + b_{11,1}w_{4,1} \\ b_{0,2}w_{1,2} + b_{1,2}w_{1,3} + \ldots + b_{11,2}w_{4,1} \\ b_{0,3}w_{1,2} + b_{1,3}w_{1,3} + \ldots + b_{11,3}w_{4,1} \\ b_{0,4}w_{1,2} + b_{1,4}w_{1,3} + \ldots + b_{11,4}w_{4,1} \\ b_{0,5}w_{1,2} + b_{1,5}w_{1,3} + \ldots + b_{11,5}w_{4,1} \end{pmatrix} \\
&= \begin{pmatrix} A_{0,0} + A_{1,0} + \ldots + A_{11,0} \\ A_{0,1} + A_{1,1} + \ldots + A_{11,1} \\ A_{0,2} + A_{1,2} + \ldots + A_{11,2} \\ A_{0,3} + A_{1,3} + \ldots + A_{11,3} \\ A_{0,4} + A_{1,4} + \ldots + A_{11,4} \\ A_{0,5} + A_{1,5} + \ldots + A_{11,5} \end{pmatrix} \\
&= \begin{pmatrix} \sum_{i=0}^{(N-1)!-1} A_{i,0} \\ \sum_{i=0}^{(N-1)!-1} A_{i,1} \\ \sum_{i=0}^{(N-1)!-1} A_{i,2} \\ \sum_{i=0}^{(N-1)!-1} A_{i,3} \\ \sum_{i=0}^{(N-1)!-1} A_{i,4} \\ \sum_{i=0}^{(N-1)!-1} A_{i,5} \end{pmatrix}
\end{aligned}
\tag{2}
$$

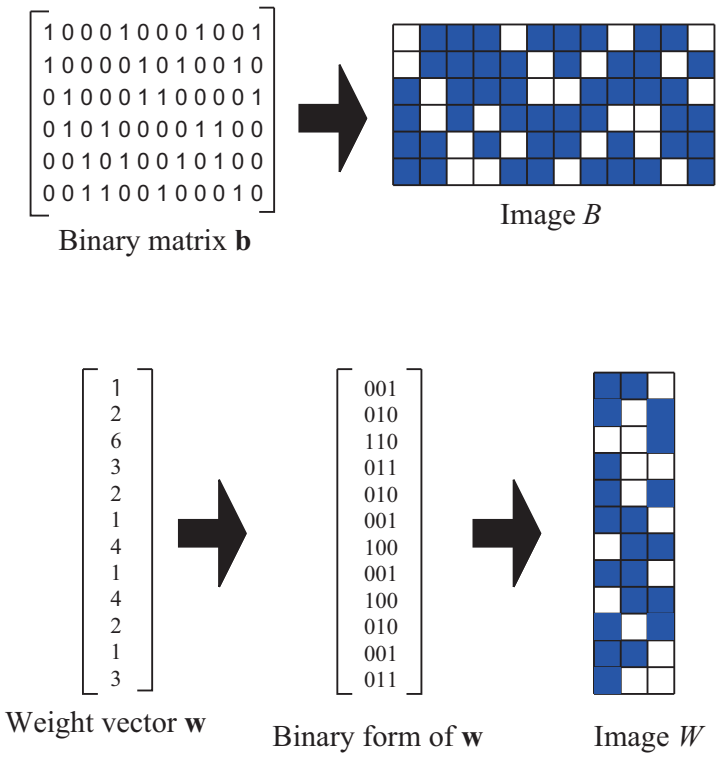

Figure 2. Image representation of the binary matrix **b** and the weight vector **w**.

Here the matrix **b** has $n(n-1) \times n-1$ of elements. Also, the dimensions of the weight vector **w** are $n(n-1)$.

In Ref.,[4] this mathematical procedure is implemented on optical hardware. This optical solution is based on a joint transform correlator. In the solution, the contents of the binary matrix **b** and the weight vector **w** are displayed on a spatial light modulator. Power spectrum of patterns displayed on the SLM is obtained with an optical system for the Fourier transform. Obtained data is input to the optics for the Fourier transform and optical signals on the output plane are measured. The shortest tour can be derived by analysis of the measured patterns.

2.2. Procedure of SIMD pattern processing for the TSP

In our method, processing described in the Eq. (2) is executed in accordance with a sequence of SIMD pattern processing for 2D images. As described in Fig. 2 (b), first, the binary matrix **b** and the weight vector **w** are converted into 2D binary images, respectively. Considering that $b_{i,j}$ is the pixel value at (i,j) on the image B, $b_{i,j}$ is determined by the (i,j) element in the matrix **b**. On the other hand, patterns at j'th row on the image depend on bit patterns of j'th element in vector **w**. In this image, the left side and the right one of pixels show the MSB (Most significant bit) and the LSB (Least significant bit) of the bit patterns, respectively.

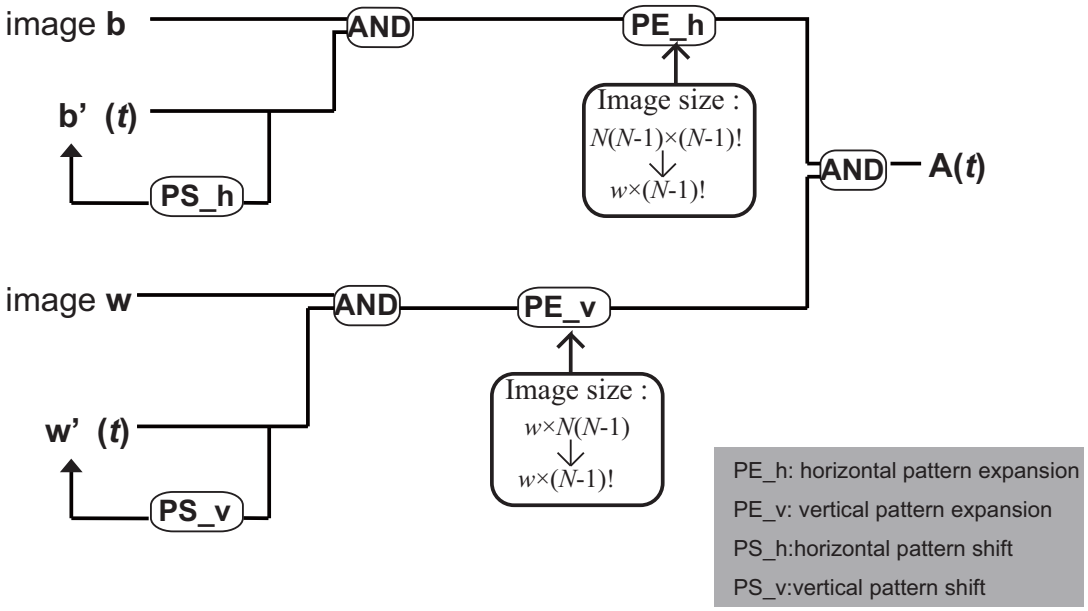

Figure 3. Block diagram of a sequence for the process to derive $A_{i,j}$.

From these two images, a set of $A_{i,j}$ is obtained by a sequence of pattern processing. Figure 3 describes a block diagram of the sequence. In the diagram, $b'(t)$ and $w'(t)$ are auxiliary images and these images are updated by pattern shift. Figure 4 shows data flows of $b'(t)$ and $w'(t)$. Additionally, flow of output images $A(t)$ are indicated in this figure. Patterns at jth row on $A(t)$ correspond with $A_{t,j}$.

Next process is summation shown in Eq. (2). The additional gate presented in our previous works[1,6] is used for the process. Figure 5 shows a block diagram for the summation process. From the figure, $C(T)$ shows an output image at $t = T$ At $t = T$, the bit patterns represented as row datum on $A(T)$ and that on $C(T-1)$ are added and the results of addition are output as row datum on $C(T)$. Considering overflow in the addition, horizontal pixel size of $C(T)$ is set to as twice as that of $A(T)$. As a result, patterns at jth row on $C(N(N-1))$ correspond with $\sum_{i=0}^{(N-1)!-1} A_{i,j}$.

Last process is extraction of the minimum tour. Figure 6 shows an examle of the results in the above summantion process. From Fig. 6(a), l_2 is considered to be the desired tour as described in Fig. 7 (b). Figure 7 shows the contents of the process.

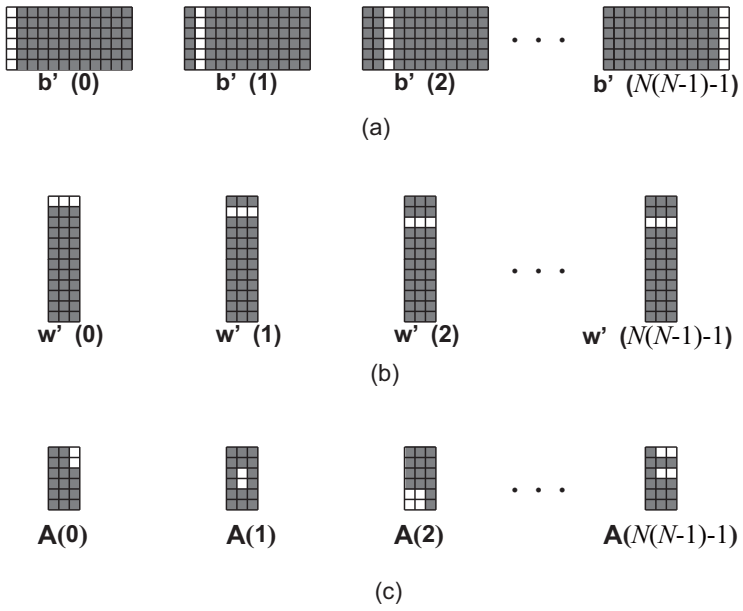

Figure 4. Data flows of (a) $b'(t)$, (b) $w'(t)$, and (c) $A(t)$ in the process shown in Fig. 3

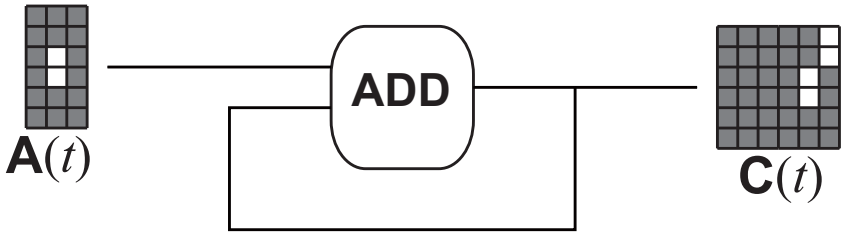

Figure 5. Block diagram of summation process

3. IMPLEMENTATION

To verify effectiveness of a GPU, our method for the TSP is implemented on a CPU and a GPU. The specifications of used CPU and GPU are summarized in Tables 1 and 2, respectively.

We implement our solution for the TSP on both the CPU and the GPU. Processing time to complete the TSP is measured to estimate usefulness of GPU implementation. Figure 8 shows the results of measurement. In the graph, relations between the processing time and the number of cities are plotted. Additionally, we implement our TSP based on a scheme for 2D SIMD processing with image compression and estimate the implementation.

From the results, it is verified that the GPU can solve the problem in shorter time than the CPU. Then, the larger the number of cities is, the more effective the GPU implementation is. However, processing time of the GPU implementation with image compression is longer than that of the GPU implementation with simple pattern processing. Also, processing time of all implementation grows exponentially in the number of cities. Therefore, we should improve a procedure based on the image compression for more effective implementation.

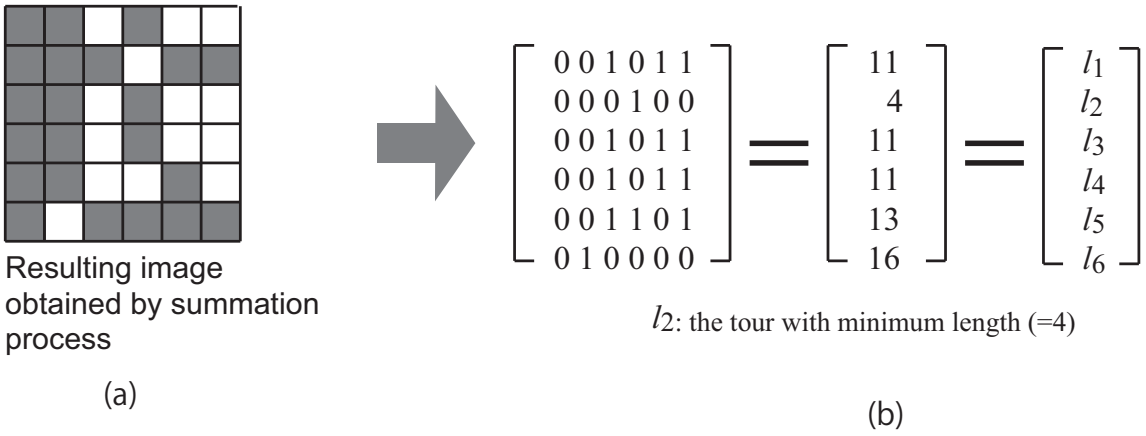

Figure 6. (a) Example of resulting images in summation process and (b) interpretation of the image of (a).

Table 1. Specifications of the CPU

CPU	Athlon X2 4600+ (2.4GHz)
RAM	3.25GByte
OS	Windows Vista
Environment	Visual Studio .net
Language	C++

Table 2. Specification of the GPU

GPU	NVidia GeForce 8800
Global Memory	10 MByte
Shared Memory	16KByte
Number of threads/a stream processor	512
Number of stream processor	112
Environment	CUDA 2.0

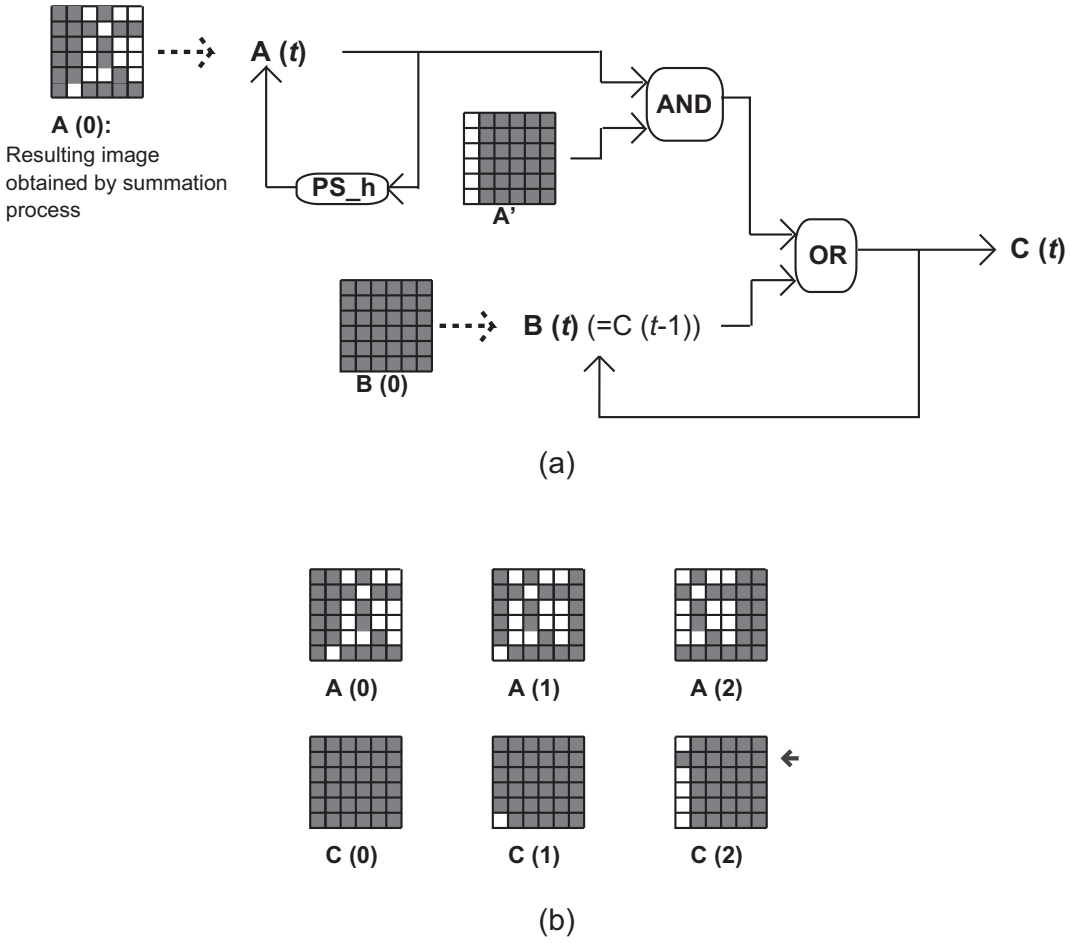

Figure 7. (a) Block diagram of the process to extract the minimum tour and (b) data flows of an input image $A(t)$ in the process.

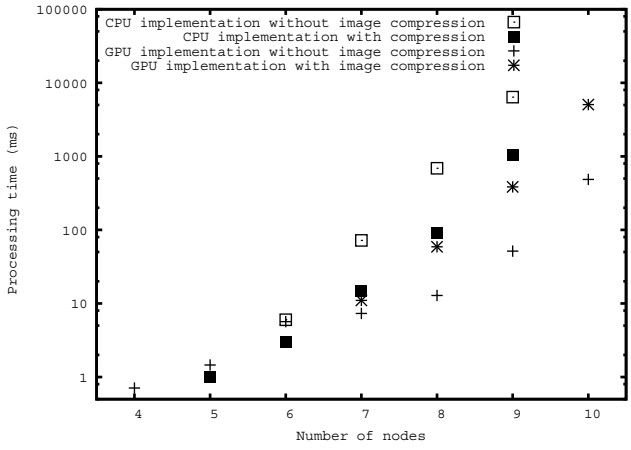

Figure 8. Relations between processing time and implementation.

4. SUMMARY

We have reported implementation of a solution for the TSP. This solution is based on SIMD pattern processing for 2D image data. In the processing, some logical operations between neighborhood pixels are utilized as elements of processing. Details of a procedure for the TSP is described in this paper.

We have implemented the procedure on both a CPU and a GPU, respectively. Processing time of both implementation are compared. As results of comparison, usefulness of the GPU in the TSP has been clarified.

REFERENCES

1. K. Nitta, Y. Tado, O. Matoba, and T. Yoshimura, "A method for factorization by means of digital optical computing and image compression," *SPIE.* **6695**, p. 66950B, 2007.
2. K. Nitta, S. Ohta, and O. Matoba, "Implementation of the tsp problem based on pattern proecssing with a graphics processing unit," *SPIE* **7442**, pp. 7442–14, 2009.
3. T. Haist and W. Osten, "An optical solution for the traveling salesman problem," *Optics Express* **15**, pp. 10473–10482, 2007.
4. N. T. Shaked, S. Messika, S. Dolev, and J. Rosen, "Optical solution for bounded np-complete problem," *Appl. Opt.* **46**, pp. 711–724, 2007.
5. N. Masuda, T. Ito, T. Tanaka, A. Shiraki, and T. Sugie, "Computer generated holography using a graphics processing unit," *Optics Express* **14**, pp. 603–608, 2006.
6. K. Nitta, Y. Tado, O. Matoba, and T. Yoshimura, "Simulation method for quantum algorithm using optical array logic," *in Technical digest in OSA Topical Meeting on CD-ROM* **JWB8**, 2005.