A Parallelized OSPF Weight Setting Scheme based on a Genetic Algorithm for Multi-Core CPUs

Ko Kikuta, Satoru Okamoto, Eiji Oki, and Naoaki Yamanaka

***Abstract*—Open Shortest Path First (OSPF) is one of the most widespread routing protocols in the world. In OSPF networks, routers calculate the paths for every traffic demand, based on weight values that are configured in advance. OSPF Weight Setting (OSPF-WS) is an NP-hard search problem; find the set of weights that maximizes network utilization. Internet Service Providers (ISPs) are facing the challenge of solving OSPF- WS within practical time and find the best weight set for the efficient use of the network. One heuristic approach, a scheme based on the Genetic Algorithm (GA), has been reported to offer fast solution of OSPF-WS. This scheme identifies good solutions comparable with the output of the conventional Integer Linear Programming (ILP) scheme. However, its calculation cost is still excessive for larger networks, thus higher processing performance is required. Unfortunately, the processing speed of single processing cores has become saturated, and the recent trend is a shift to multi-core processors. To best utilize the performance offered by these processors, the algorithm should be redesigned and suitably parallelized for multi-core CPUs. This paper redesigns the scheme of OSPF-WS with GA to create a parallelized algorithm with much lower computation overhead. Its performance is evaluated on a 16-core Intel Xeon processor and the result is a roughly 13 fold faster calculation speed than the original algorithm on a single-core CPU. This result shows the potential of further speedups with larger scale parallel processing units such as the GPGPU.**

***Index Terms*—OSPF-WS, Traffic Engineering, Genetic Algorithm, Parallelization.**

I. Introduction

COMPUTER networks such as the Internet have become one of the infrastructures necessary for people nowadays. Internet Service Providers (ISPs) are responsible for the exchange of customer's data while satisfying the QoS requirements specified in the Service Level Agreements (SLAs). For efficient operation of limited network resources, Traffic Engineering (TE) is the most important challenge for ISPs.

K. Kikuta, S. Okamoto and N. Yamanaka are with the Department of Information and Computer Science, Faculty of Science and Technology, KEIO University, 3-14-1 Hiyoshi, Kohoku-ku, Yokohama, 223-8522 Japan.

E. Oki is with Graduate School, Faculty of Science and Technology, Keio University, and Dept. of Communication Engineering and Informatics, Graduate School of Informatics and Engineering, The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu-shi, Tokyo 182-8585, JAPAN.

Manuscript received August 20, 2013; revised August 20, 2013.

Open Shortest Path First (OSPF) [1] is the most widely-spread intra-domain routing protocol in the Internet. In an OSPF network, the routes for the data traffic are determined by link weights. Each link is assigned a link weight, a 16-bit integer value ranging from 1 to 65535. Every router exchanges its weight value throughout the Autonomous System (AS) and a shortest-path is calculated for each source-destination pair considering the weight value as the virtual distance of a link. A larger link weight value makes the link unlikely to be the shortest-path, so all traffic flows are determined by the setting of weights. Additionally, routers divide traffic flow equally for Equal Cost Multi Path (ECMP) if there are multiple shortest-paths available. If TE is to realize the efficient use of the network, which may involve ECMP, weight setting is crucial task for ISPs. As a leading router vender, Cisco recommended the capacity-inverse setting of link weights [2]. When the traffic demand is estimative, this approach does not appear to be the best way.

If the network is represented as directed graph $G = (N, E)$, and the traffic demand is given by matrix $D = \{D_{st} : s \in V, t \in V\}$ (s and t denote a source node and a destination node, respectively, and D_{st} is a traffic demand from s to t), weight set $W = \{w_1, w_2, ..., w_{|E|}\}$ determines the routes for all traffic flows, and thus the congestion ratio f_e/C_e, on any edge e (f_e denotes the total of the traffic flows on edge e, C_e denotes the capacity of edge e). Minimizing $L = max_{e \in E} (f_e/C_e)$, the congestion ratio on the most congested link, maximizes network utilization according to [3], OSPF Weight Setting (OSPF-WS) is the problem of identifying W that minimizes L and maximizes network utilization under given demand matrix D. This problem is known to be NP-hard so computation cost instantly becomes excessive when network size increases. ISPs are forever trying to solve OSPF-WS within a practical time period in the face of this complexity.

If OSPF-WS is to be practical, the calculation time must be limited. If a traffic demand is updated, the current weight set may not be suitable for the updated traffic demand. There- fore, a new weight set needs to be calculated in a dynamic manner to follow the traffic demand changes for the given network topology. Note that the network topology may also be optimized as in the case of a virtual network such as an optical-path network, which consists of optical paths that can be dynamically setup and released according to traffic demand changes. In this

situation, multi-layer optimization is required so as to configure both the virtual network and OSPF weight setting. To achieve multi-layer optimization, an interactive computation between the virtual network optimization and the OSPF weight optimization is performed. Since the OSPF weight optimization for a given virtual network topology is required iteratively, it should be performed extremely rapidly.

Several calculation schemes for OSPF-WS have been introduced [4]. The two main types of solutions: Integer Linear Programming (ILP) and heuristic search. In the former, ILP, the network optimization problem of OSPF-WS is transformed into a mathematical ILP problem to allow application of the ILP solver. ILP was originally designed to find the optimal solution by exhaustive search, but also to find near-optimal solutions as preliminary results of that search. The ILP solver can finally achieve a good solution, but it takes a long time, even for the first feasible solution. This time is not short enough for dynamic weight setting unless the network is small. ILP computation is not reliable since there is no assurance that even the first solution will be available within a practical time. Therefore, ILP is inadequate for dynamic weight setting.

On the other hand, heuristic search ignores the optimal solution to secure close-to-optimal solutions in much shorter times. Most OSPF-WS schemes employ metaheuristics. One of them is based on Genetic Algorithm (GA) proposed by Ericsson et al. [5]. While this scheme does not guarantee the accuracy due to its use of stochastic search, the solution is, on average, comparable to ILP. Additionally, the best solution is always available at any moment during the calculation in this scheme. Thus the scheme is preferred for dynamic weight setting. However, its calculation costs are still huge when the network size is large. Its support of larger networks revolves around the processing performance.

Recent CPU venders are focusing on the development of multi-core CPUs since clock speed has reached a technical limitation [6]. Unlike over-clocking the CPU, providing more processing cores does not directly accelerate the original algorithms designed for single-thread processing. To utilize the full performance of parallel core technology, the scheme must be redesigned at the algorithm level. No parallel version of OSPF-WS has been reported to date.

This paper parallelizes OSPF-WS for multi-core CPUs and so enables TE by dynamic weight setting. The proposed scheme, based on [5], is evaluated on a 16 core Intel Xeon Processor. It is shown to be about 13 times faster than the original algorithm designed for single-core implementation.

II. Related Research

A. OSPF-WS with Integer Liner Programming

ILP is a mathematical approach to the optimization problem of minimizing (or maximizing) an objective function under a given constraint. To solve OSPF-WS with ILP, the problem, one objective function and a number of constraint conditions, is first expressed as equations and inequalities, where all functions must be linear and able to include integer variables.

A typical ILP scheme is proposed in [7]. In this scheme, OSPF-WS is expressed as follows, (the expression are slightly modified and comments are added for understanding).

The OSPF network is expressed by directed graph $G = (V, E)$ where a node denotes a router and an edge denotes a link. Edge $e \in E$ has bandwidth capacity of c_e, demands are given as matrix D and element D_{st} denotes the traffic demand from source node $s \in V$ to $t \in V$, $Vd \subseteq V$ is a set of destination nodes.

Decision Variables:

f_e^t : amount of traffic flow to destination t on edge e
x_e^t : binary variable denoting if flow f_e^t exists or not
f_v^t : amount of split flow to node t on node v
w_e : weight value of edge e
d_v^t : shortest distance from node v to destination t
L : the maximum link load over all edges

Flow Conservation Constraints:

$$\sum_{e:e=(-,t)} f_e^t = \sum_{v \in V} D_{vt} \quad \forall t \in V_d \tag{1}$$

$$\sum_{e:e=(-,v)} f_e^t - \sum_{e:e=(v,-)} f_e^t = -D_{vt} \quad \forall v \in V \backslash \{t\} \quad \forall t \in V_d \tag{2}$$

Equations (1) and (2) define the relationship between incoming and outgoing flows. Eq. (1) ensures that flows are terminated at their destination nodes, and Eq. (2) ensures flow conservation since the difference between incoming flows and locally dropped flows must equal the flows passed on to other nodes.

Flow Splitting Constraints :

$$M x_e^t - f_e^t \geq 0 \quad \forall e \in E, \quad t \in V_d \tag{3}$$

$$f_v^t - f_e^t \geq 0 \quad \forall e = (v, -) \in E, \quad \forall v \in V \backslash \{t\}, \quad \forall t \in V_d \tag{4}$$

$$f_v^t - f_e^t \leq M(1 - x_e^t) \quad \forall e = (v, -) \in E, \quad \forall v \in V \backslash \{t\}, \quad \forall t \in V_d \tag{5}$$

Inequality (3) ensures that only when the binary variable x_e^t is equal to 1 can f_e^t be a non-zero value, provided M is a very large number, (This M can be set to the sum of D_v^t). Hence, if f_e^t is non-zero, x_e^t must be 1. In this case, the inequalities (4) and (5) ensure that f_e^t must be equal to f_v^t. On another front, if both x_e^t and f_e^t are zero, inequalities (4) and (5) have no effect on f_v^t. Therefore, f_e^t must be f_v^t or zero according on x_e^t. In this way, equal flow splitting is realized by virtual value f_v^t .

Feasible Distance Label Constraint :

$$d_v^t + w_e - d_u^t \geq 0 \quad \forall e = (u, v) \in E, \quad \forall t \in V_d \tag{6}$$

$$d_v^t + w_e - d_u^t \leq M(1 - x_e^t) \quad \forall e = (u, v) \in E, \quad \forall t \in V_d \tag{7}$$

$$M(d_v^t + w_e - d_u^t) \geq 1 - x_e^t \quad \forall e = (u, v) \in E, \quad \forall t \in V_d \tag{8}$$

The mechanism of inequalities (6), (7) and (8) is somewhat complex. If edge *e* has a flow to destination *t*, x_e^t equals 1 (as shown before), then $d_v^t + w_e$ equals d_u^t according to inequalities (6) and (7). Since every neighbor node pair at the ends of edge *e* along the path has this relationship, d_u^t must be sum of w_e on the path. On the other hand, the other edges out of paths that have x_e^t equal to zero must not be the shortest path since $d_v^t + w_e$ is greater than d_u^t according to the inequalities (8). In this way, if a flow exists and x_e^t equals to 1, it is ensured that the edge *e* is on the shortest path.

$$\sum_{t \in Vd} f_e^t \leq Lc_e \quad \forall e \in E, \quad \forall t \in V_d \tag{9}$$

The last inequality (10) ensures that the maximum value of the link congestion ratio is less than *L*. Since *L* is to be minimized, *L* indicates the maximum value of link congestion ratio.

$$\begin{aligned} &\text{minimize} \quad L \\ &s.t. \quad x_e^t = \{0,1\}, \quad f_e^t \geq 0, \quad f_v^t \geq 0, w_e \in \mathbb{N}, \quad d_v^t \in \mathbb{N}, \quad L \geq 0 \end{aligned}$$

The ILP problem can, as expressed above, be tackled by ILP solver software. CPLEX was developed by IBM and has become one of the most popular commercially available ILP solvers, and used in [7]. CPLEX reads the problem as the input of the objective function and multiple equations and inequalities, then starts its run. The functions are analyzed and a solution is located. In most cases, CPLEX derives better solutions than naive heuristic search as explained later.

ILP solvers output sets of decision variables, not just a weight set. That is, the solution space of ILP includes infeasible solutions which do not satisfy one or more restrictions. During the solution search, the ILP solvers try to minimize the objective function, L. When all variables satisfy all restrictions, the solver can output the first feasible solution, which includes a weight set. Unfortunately, the time taken to reach the first solution becomes exponentially large if network size is large. In this case, the ILP solver fails to output any weight set in practical calculation time.

B. OSPF-WS with Genetic Algorithm

On the other hand, heuristic search finds solutions based on experience rather than identifying the optimum solution. For OSPF-WS, some heuristic search schemes based on metaheuristics are reported in [4]. B. Fortz et al. [8] presented the Local Search algorithm and M.H.Sqalli et al. [9] presented Simulated Annealing; Ericsson et al. used the Genetic Algorithm [5] to solve OSPF-WS.

GA is a metaheuristic proposed in [10] that imitates the principle of natural selection. GA has been applied to various optimization problems even in the networking area, such as multicast routing [11] and regenerator placement [12]. In GA, each solution candidate (called individual) is expressed as a gene, and it forms a group (called population). In the population, evolutionary events such as inheritance, mutation, selection and crossover are performed iteratively (each itera- tion is called a generation). In each generation, the individuals are evaluated by some fitness metric. The individuals that have, or potentially have, better fitness are generated by evolution. A gene is the data

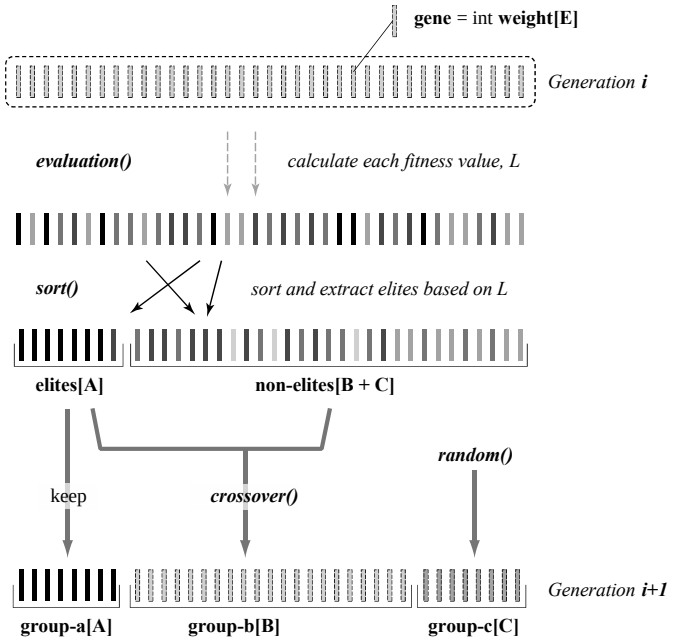

Fig. 1. Erricsson's scheme of GA for OSPF-WS

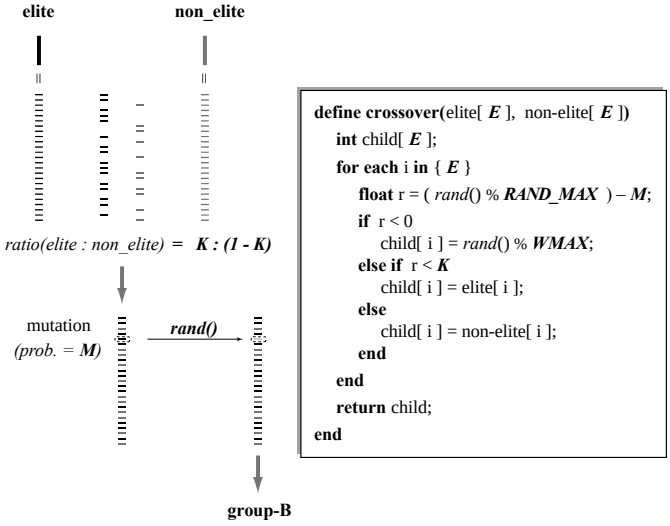

Fig. 2. Crossover Function

that encodes the property of the individuals. As the gene evolves in every generation, the best solution is improved. More details such as gene encoding, function to calculate fitness, and the procedure of evolution depend on the scheme. These procedures govern GA performance.

Figure 1 shows Ericsson's GA for OSPF-WS [5]. In this scheme, a gene is directly encoded as an OSPF-WS weight set, and the fitness value is L. The population size per generation is

constant at *P*, and all initial genes are given by random numbers. Each generation is associated with two routines; one evaluates all individuals and the other creates new genes by evolving the current genes. In the latter, first all genes are sorted based on fitness value and categorized as either an elite genes or a non-elite gene. The elite genes are preserved as group-A genes of next generation. The group-B genes are processed by a crossover function where randomly chosen pairs of elite and non-elite genes are recombined to yield new genes. Group-C genes are created by setting random numbers. If group size is expressed as *A*, *B* and *C*, the total population $P = A + B + C$ (that is, number of elite genes is *A*, number of non-elite is $B + C$). After all genes have been processed or generated, the scheme proceeds to the next generation.

Figure 2 shows the crossover function generating group-B genes which is characteristic of the Genetic Algorithm. To generate a new gene, one elite gene and one non-elite gene are chosen at random and recombined. Each value in the array of the weight set of the new gene is selected from the elite gene or the non-elite gene at the same position. The selection is biased by parameter *K*, so the elite gene value is selected with probability of *K*, the non-elite with probability $(1 - K)$. Furthermore, the rare event of mutation (replacement by a random value) is determined by parameter *M*. A pseudo code that performs weight selection with one random number is shown in the right part of Fig. 2. The genes generated by this crossover potentially have better fitness value. If some of them are better than some the current group-A entries, they replace the group-A entries in the next generation. Evolution is performed in this manner.

The important issue with algorithm acceleration on a multi-core CPU is represented by Amdahl's law [14]. According to this law, the speedup possible with parallelization is limited to the sequential fraction of the whole process. When a scheme is parallelized on *N* processing elements with a fraction of *x* of total running time, the theoretical ratio of processing speed $S(N)$ accelerated by parallelization is given as follows.

$$S(N) = ((1 - x) + x/N)^{-1} \qquad (10)$$

More specifically, parallelizing 90 percent of the whole process ($x = 90$) on 16 cores ($N = 16$) achieves a 6.4 times speedup. If a 10 times speedup is needed, 96 percent of the whole process should be parallelized. Furthermore, the computational overhead of parallelization such as synchronization latency and processing time of thread management is another factor limiting the speedup. For effective parallelization, the scheme must be parallelized globally with small computational overhead.

III. Parallelization of OSPF-WS GA

In this research, the Ericsson scheme for OSPF-WS with GA [5] is parallelized at the algorithm level. First, the con- ventional serial version of the scheme is explained, then the proposed parallelization version is introduced.

A. The serial algorithm of OSPF-WS GA

In the algorithm, both the evaluation routine and the evolution routine are processed in a large iteration loop for each generation. The evaluation routine consists of an iteration of the evaluation() function. In the evaluation() function, all routes for demanded pairs are calculated by Dijkstra's algorithm with given weight set, which is initially taken to be link distance. Second, flow size f_e on every link on the network is calculated by assignment of all demands to all links along the route. Finally, the congestion ratio of the most congested link is returned as *L*. This function of evaluation() is processed iteratively for group-B and group-C genes in the evaluation routine. In the subsequent evolution routine, the population is sorted based on fitness value *L* and categorized into elites and non-elites. In order to reduce the cost of sorting, the data element to be sorted is a pointer to gene data, which is structured with fitness value data. Hence, the population is managed by an array of this structure. After sorting, the crossover() function is processed iteratively to generate new genes of group-B. The crossover() function chooses pairs of elite and non-elite genes by using random numbers and then a new gene is generated. Since all genes in group-B and group-C are potentially referred to in each iteration of the crossover() function, these genes data are not able to be rewritten until all iterations of crossover() function have been processed. For this reason, buffers for new genes in

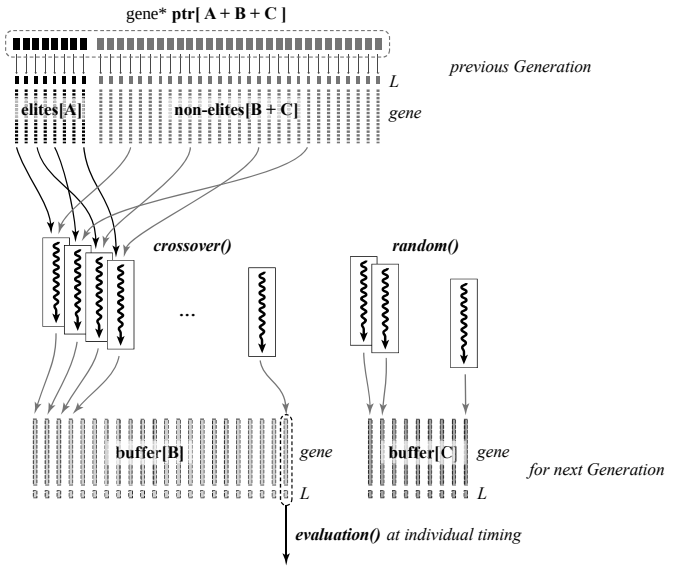

Fig. 3. Data structure for Parallelization

group-B are introduced, and new genes are written in the buffer. After all iterations of crossover() function are processed, the pointers of all genes in group-B are swapped. Finally, all genes in group-C are replaced by random numbers as per the initial genes. As all genes for the next generation are ready, the scheme proceeds to the next generation.

B. proposed parallelization of OSPF-WS GA

This parallelization is processor-independent, so it makes no assumption as to the number of processing elements. The main

strategy is parallelization of the iterative process for each gene.

The scheme for OSPF-WS with GA is parallelized in two steps. In the first step, the evaluation routine is parallelized. This routine has only one iterative loop of the evaluation() function. Both input data and output data for each evaluation() function are independent, so this function can be processed in parallel without any change. Then each iteration is parallelized and processed by multiple threads. The number of threads equals the

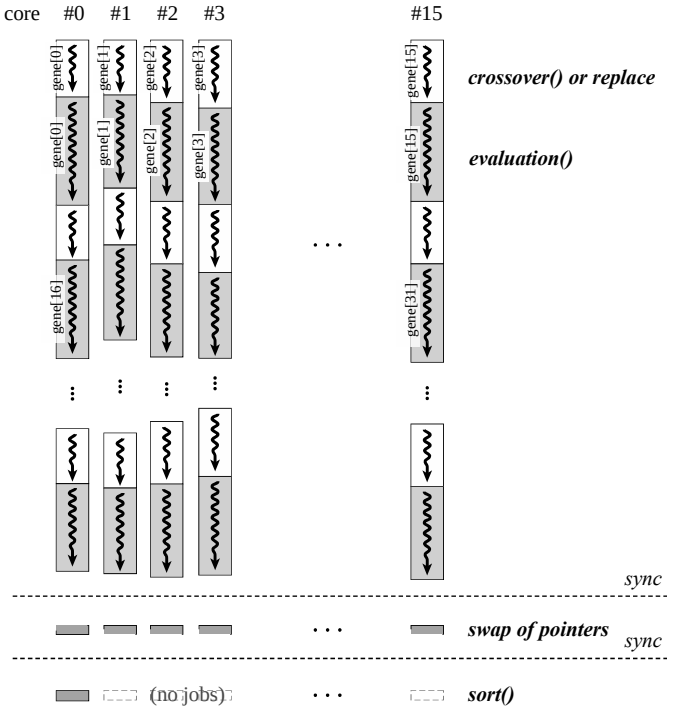

Fig. 4. Image of Thread Graph in Proposal

number of cores and task assignment to each thread is performed dynamically. Since the evaluation routine occupies the greatest part of the total process in each gener- ation, parallelization of evaluation achieves the most effective speedup.

The second step is global parallelization, where the evolution routine is also parallelized. In this evolution routine, random numbers are frequently used so the random number table is parallelized first. Every execution of random() involves modification of the table. If the table is shared, the speed of accessing the table is low due to data coherency. Our solution is to use as many random tables as there are threads.

The first process of evolution is to sort the genes. This is not parallelized since its processing cost is negligible. The remaining processes of evolution are three iteration loops for crossover() function for group-B, pointer swap for group- B, and generation of group-C. If these iteration loops are parallelized individually, three parallel sections are created in this routine. To reduce the computational overhead, our proposed scheme minimizes the number of parallel sections. First, a buffer is introduced for the generation of group-C genes as shown in Figure 3. This allows group-C genes to be generated without rewriting the non-elite genes referred to by group-B. All group-B and group-C genes can be generated in parallel in the same parallel section. Second, a buffer is also prepared for L data (and is structured with the gene data buffer for better management) for each of group-B and group-C genes. This buffer allows a generated gene to be evaluated immediately and independently without waiting for pointer swap. Then, the parallel section processing group-B and group-C genes is merged with the parallel section of the evaluation routine in the next generation. This is, the order of the evaluation routine and evolution routine in the large iteration loop is inverted. The pointer swapping is performed for both genes and L values of all group-B and group-C genes. Overall, except for the pointer swapping section, three parallel sections are merged into one as shown in Figure 4. Thus the proposed scheme consists of only two parallel sections in total and so reduces the parallelization overhead.

IV. Performance Evaluation

This section evaluates the performance of the proposed algorithm. For this evaluation, OSPF-WS problems were created by combining random traffic demands with randomly generated networks based on Waxman's model [15]. All problems were solved by ILP [7], a serial scheme of GA by Ericsson [5], and the proposed parallel GA scheme. GA parameters such as size of group-A, group-B, group-C were $A = 300$, $B = 3000$, $C = 300$, while $K = 0.5$, $M = 0.01$. For ILP, the CPLEX solver was used. All calculations were performed on a 16-core Intel(R) Xeon(R) CPU (E5-2687W 3.10GHz).

A. Speedup by Parallelization

First, the speedup achieved by parallelization was evalu- ated. This proposed parallel scheme basically uses the same procedure as Ericsson's GA [5]. Therefore, the number of generations processed per second is a fair measure of the performance of each scheme. For the 10 sets of 15 node networks and 10 sets of traffic demands, 100 generations were measured for both schemes and averaged as units of 5 runs.

TABLE I
Speedup by Parallelization

	Serial	Parallel (partial)	Parallel (global)
time(sec)	3.068	0.375	0.233
speedup	**1.0**	**8.17**	**13.16**
IPC	1.14	0.82	1.07
Instructions (bn)	13.25	16.63	13.59

The results of speedup are shown in Table. I. This table lists the results of serial scheme, partial parallel scheme (only evaluation routine is parallelized in the first step), and the proposed globally parallelized scheme. The speedup value is normalized against the serial scheme. Partial parallelization achieved a 6.85 times speedup while the scheme (global parallelization) achieved a 13.16 times speedup. This speedup corresponds to about parallelization of 99 percent. The reason why the proposal failed to reach a 16 times speedup is as follows. First, using the same data on multiple cores reduces the validity of cached data. Accordingly, the metric of Instructions Per Second must be degraded from 1.14 to 1.07 as shown in the table. Next, to manage the parallel session, extra instructions such as

synchronization or assignment of tasks are needed, as also shown in the table.

B. *Speedup by Parallelization*

Next, the minimized L values output of both schemes are evaluated. For this evaluation, 200 random demand pairs were given for 10 random networks, each with 50 nodes. The L value directly indicates the efficiency of network utilization. The calculation time is 60 sec, which assumes dynamic weight setting.

TABLE III
L COMPARISON AGAINST OTHER SCHEMES

network	Minhop	Cisco	ILP	Prop.	ILP/Prop.
#0	0.0833	0.3333	--	0.0244	--
#1	0.0690	0.1563	0.0500	0.0500	1.00
#2	0.0682	0.1667	--	0.0235	--
#3	0.0857	0.1333	0.0333	0.0333	1.00
#4	0.0761	0.0870	0.0500	0.0500	1.00
#5	0.1053	0.1333	0.0187	0.0208	0.90
#6	0.0714	0.2353	0.0200	0.0250	0.80
#7	0.1023	0.0909	--	0.0215	--
#8	0.0606	0.1429	0.0257	0.0262	0.98
#9	0.1071	0.1875	0.0714	0.0313	2.28

Table. II shows L values output by the conventional serial and proposed parallel scheme. Even though network size and number of demands are the same, the different networks yield different problems and thus different optimal values. The proposed scheme reduces L from 1.2 to 1.7 times, on average 1.46 times smaller (better) than the serial scheme.

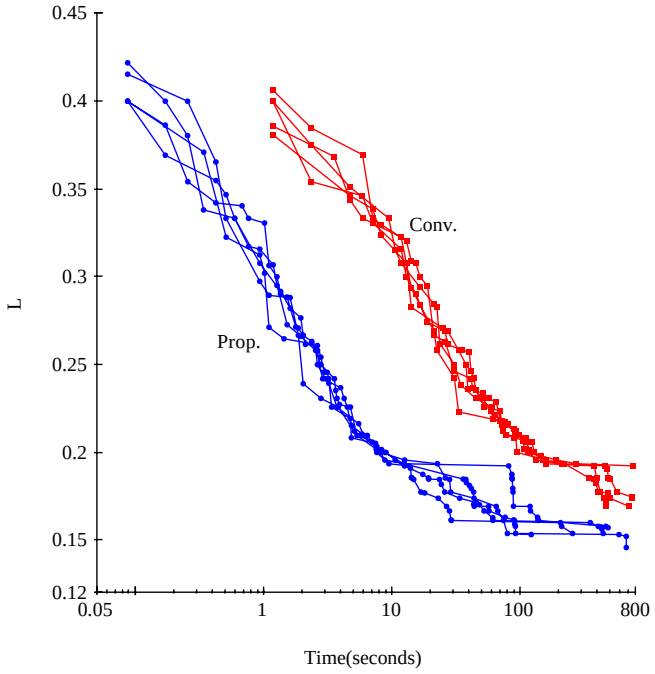

Fig. 5. Serial vs. Parallel

Figure 5 shows the time variation in calculated L value which is one of above results (taken for network #1). The horizontal axis is logarithmic. Although stochastic search disperses the plots slightly, it reduces L in most part.

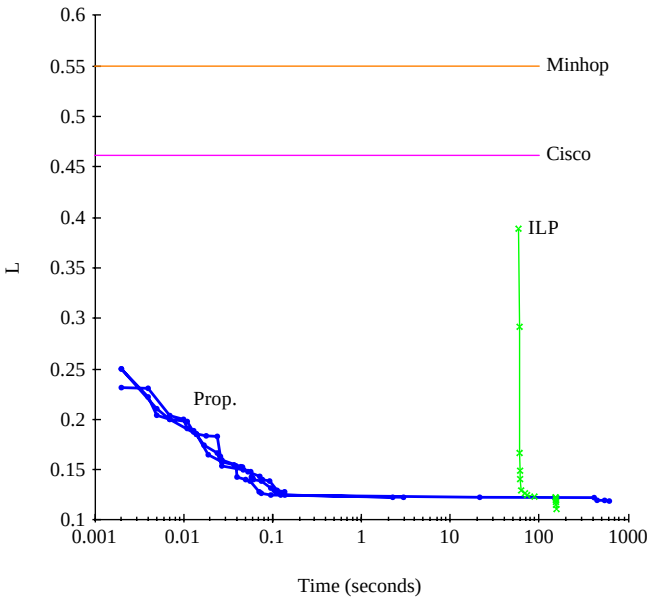

Fig. 6. Proposed vs. ILP and others

TABLE II
COMPARISON OF L WITH SERIAL SCHEME

Network	Serial	Parallel	Serial/Parallel
#0	0.2512	0.1538	1.63
#1	0.2268	0.1724	1.32
#2	0.2390	0.1982	1.21
#3	0.2710	0.1697	1.60
#4	0.2323	0.1403	1.66
#5	0.2467	0.1967	1.25
#6	0.2163	0.1262	1.71
#7	0.1893	0.1179	1.61
#8	0.2844	0.2333	1.22
#9	0.2779	0.2000	1.39
Average			1.46

C. *Comparison with other scheme*

Finally, the effectiveness of the proposed scheme for OSPF-WS is shown as a comparison with other methods. Table III shows OSPF-WS output by the proposed scheme (shown as Prop.), ILP with CPLEX, and other constant weight setting methods. Also, Figure 6 shows the time variation of L on network #2. As constant setting methods, Minhop sets all weights to 1, Cisco sets them to the inverse of bandwidth. To achieve results from ILP, network size was set at 20 nodes and calculation time at 800 sec.

As shown in the figure, since network size is small, the proposal converges rapidly. On the other hand, ILP needs a long time to output the first solution. As shown in the table, ILP could not find any solution in 800 sec for some networks (#0, #2, #7). As the network becomes larger, more problems become insoluble by ILP. The L value of the proposed scheme is small enough compared to Minhop or Cisco. In comparison with ILP, the proposal generally yields the same solution. Even if the ILP

solution is better (smaller L), the different is not significant.

V. Conclution

OSPF-WS is an NP-hard problem so its computation cost is excessive for any practical network size. Although ILP can finally achieve a good solution, it fails to support dynamic weight setting because it takes too long to identify the first feasible solution. GA is preferred for dynamic weight setting but calculation costs are huge if network size is large. Its support of larger networks depends on the adoption of recent multiple CPU core technology by parallelization at the algorithm level and careful consideration of parallelization bottlenecks such as computational overhead and the sequential fraction.

This paper introduced a parallelized OSPF-WS scheme for multi-core CPU implementation as an enhancement of Ericsson's GA-based scheme [5]. This parallelization is performed globally in both the evaluation routine and the evolution routine, and computation overhead is minimized. The proposed scheme achieves a roughly 13 times speedup on a 16 core CPU. The solutions of the proposed scheme are comparable with those of the ILP scheme, and it remains feasible even for larger networks such as those with 50 nodes.

Acknowledgment

This work was supported by the Japan Society for the Promotion of Science's (JSPS) Grant-in-Aid for Scientific Reseach(A) 22240004.

References

[1] J. Moy, "OSPF Version 2," IETF Proposed Standard, RFC 2328, April 1998.
[2] T.M. Thomas II. "OSPF Network Design Solutions." Cisco Press, 1998.
[3] Eiji Oki and Ayako Iwaki, "Load-Balanced IP Routing Scheme Based on Shortest Paths in Hose Model," IEEE Transactions on Communications, Vol. 58, Issue 7, pp. 2088-2096, July 2010
[4] Ghazala, A.A., et al., "A Survey for Open Shortest Path First Weight Setting (OSPFWS) Problem," International Conference on Information Security and Assurance(ISA), April 2008.
[5] M. Ericsson, et al., "A Genetic Algorithm For The Weight Setting Problem In Ospf Routing," pp. 299-33, vol. 6, 2002.
[6] D.A. Patterson, J.L. Hennessy, "Computer Organization and Design," Fourth Edition, Morgan Kaufmann, November 2011.
[7] M. Pioro, et al., "On open shortest path first related network optimization problems.Performance Evaluation," pp. 201-223, vol. 48, 2002.
[8] Bernard Fortz and Mikkel Thorup,"Internet traffic engineering by optimizing OSPF weights," IEEE Conference on Computer Communications (INFOCOM), pp. 519 Γ 528, vol. 2, 2000.
[9] Mohammed H. Sqalli, et al., "An Enhanced Estimator to Multi-objective OSPF Weight Setting Problem", Network Operations and Management Symposium (NOMS), pp. 240 - 247, April 2006.
[10] J.H. Holland. "Adaptation in Natural and Artificial Systems," MIT Press, 1975.
[11] Ting Lu, et al., "Genetic Algorithm for Energy-Efficient QoS Multicast Routing," IEEE Communications Letters, Vol. 17, Issue 1, pp. 31-34, January 2013.
[12] Zuqing Zhu, et al., "Using Genetic Algorithm to Optimize Mixed Placement of 1R/2R/3R Regenerators in Translucent Lightpaths for Energy-Efficient Design," IEEE Communications Letters, Vol. 16, Issue 2, pp. 262-264, February 2012.
[13] Dijkstra, E. W., "A note on two problems in connection with graphs," Numerische Mathematik, pp. 269 Γ 271, 1959.
[14] Gene M. Amdah, "Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities," Proceeding of AFIPS Conference, pp. 483 Γ 485, April 1967.
[15] B.M.Waxman, "Routing of multipoint connections," IEEE Journal on Selected Areas in Communications, vol. 6, issue 9, pp. 1617-1622., Dec 1988.

Ko Kikuta graduated from Keio University, Japan where he received B.E. degrees in applied chemistry in 2007, and M.E. degrees in electronics engineering in 2009. Since 2007, he has been researching traffic engineering for GMPLS, especially the next generation layer-2 network. He is currently researching a network control technique. He is currently a Ph.D candidate in Yamanaka Laboratory, Department of Information and Computer Science, Keio University. From 2009, he has also been a Research Assistant of Global COE (Center of Excellence) program for High-Level global cooperation for leading-edge platform on access spaces of the Ministry of Education, Culture, Sports, Science, and Technology, Japan. Since 2011 he has been a Research Fellow of Japan Society for the Promotion of Science (JSPS). Ko Kikuta is a member of IEEE Comsoc. and IEICE.

Satoru Okamoto received his B.S.,M.S, and Ph.D. degrees in electronics engineering from Hokkaido University, Hokkaido, Japan in 1986, 1988 and 1994 respectively. In 1998, he joined Nippon Telegraph and Telephone Corporation (NTT), Japan. Here, he engaged in research on ATM cross-connect sys- tem architectures, photonic switching system, optical path network architectures, and developed GMPLS controlled HIKARI router (Photonic MPLS router) systems. He lead several GMPLS related interoperability trials in Japan, such as the Photonic Internet
Lab (PIL), OIF world wide interoperability demo, and Keihanna Interoperability Working Group. From 2006, he has been an Associate Professor of Keio University. He is a vice co-chair of Interoperability Working Group of Kei-han-na Info-communication Open Laboratory. He is now promoting several research projects in the photonic network area. He received the young Researchers' Award and the Achievement Award in 1995 and 2000, respectively. He has also received the IEICE/IEEE HPSR2002 outstanding paper award. He is associate editor of the IEICE transactions and the OSA Optics Express. He is an IEEE Senior Member and an IEICE Fellow.

Eiji Oki is a Professor at the University of Electro-Communications, Tokyo, Japan. He received the B.E. and M.E. degrees in instrumentation engineer- ing and a Ph.D. degree in electrical engineering from Keio University, Yokohama, Japan, in 1991, 1993, and 1999, respectively. In 1993, he joined Nippon Telegraph and Telephone Corporation (NTT) Communication Switching Laboratories, Tokyo, Japan. He has been researching network design and control, traffic-control methods, and high-speed switching systems. From 2000 to 2001, he was a Visiting Scholar at the Polytechnic Institute of New York University, Brooklyn, New York, where he was involved in designing terabit switch/router systems. He was engaged in researching and developing high-speed optical IP backbone networks with NTT Laboratories. He joined the University of Electro- Communications, Tokyo, Japan, in July 2008. He has been active in the standardization of path computation element (PCE) and GMPLS in IETF. He has written more than ten IETF RFCs and drafts. Prof. Oki was the recipient of the 1998 Switching System Research Award and the 1999 Excellent Paper Award presented by IEICE, the 2001 Asia-Pacific Outstanding Young Researcher Award presented by IEEE Communications Society for his contribution to broadband network, ATM, and optical IP technologies, and the 2010 Telecom System Technology Prize by the Telecommunications Advanced Foundation. He has authored/co-authored four books, Broadband Packet Switching Technologies, published by John Wiley, New York, in 2001, GMPLS Technologies, published by CRC Press, Boca Raton, FL, in 2005, Advanced Internet Protocols, Services, and Applications,

published by Wiley, New York, in 2012, and Linear Programming and Algorithms for Communication Networks, CRC Press, Boca Raton, FL, in 2012. He is an IEEE Fellow.

Naoaki Yamanaka graduated from Keio University, Japan where he received B.E., M.E., and Ph. D. degrees in engineering in 1981, 1983 and 1991, respectively. In 1983 he joined Nippon Telegraph and Telephone Corporation's (NTT's) Communica- tion Switching Laboratories, Tokyo, Japan, where he was engaged in the research and development of a high-speed switching system and high-speed switching technologies for Broadband ISDN services. Since 1994, he has been active in the development of ATM-based backbone networks and systems including Tb/s electrical/Optical backbone switching as NTT's Distinguished Technical Member. He is now researching the future optical IP network, and optical MPLS router systems. He is currently a professor of Keio Univ. and representative of Photonic Internet Lab. (PIL). He has published over 126 peer- reviewed journal and transaction articles, written 107 international conference papers, and been awarded 182 patents including 21 international patents. Dr. Yamanaka received Best of Conference Awards from the 40th, 44th, and 48th IEEE Electronic Components and Technology Conference in 1990, 1994 and 1998, TELECOM System Technology Prize from the Telecommunications Advancement Foundation in 1994, IEEE CPMT Transactions Part B: Best Transactions Paper Award in 1996 and IEICE Transaction Paper Award in 1999. Dr. Yamanaka is Technical Editor of IEEE Communication Magazine, Broadband Network Area Editor of IEEE Communication Surveys, and was Editor of IEICE Transaction as well as vice director of Asia Pacific Board at IEEE Communications Society. He is an IEEE Fellow and an IEICE Fellow.