Journal of the Operations Research Society of Japan
2003, Vol. 46, No. 3, 286-305

AN IMPROVEMENT METHOD FOR THE WORKFLOW MANAGEMENT SYSTEMS BASED ON THE REALLOCATION OF FLOWS USING THE GENETIC PROGRAMMING AND ITS APPLICATIONS

Shozo Tokinaga Makoto Tominaga
Kyushu University

(Received August 12, 2002; Revised February 27, 2003)

Abstract This paper deals with an improvement method for the Workflow Management Systems (WMSs) based on the reallocation of flows using the Genetic Programming (GP) and its applications. Current WMSs are usually used in a static manner and WMSs are designed based on conventional experiences. It is also necessary to modify the system by observing the failed tasks. We propose a method for the improvement and reconfiguration of WMSs based on the GP. It is expected that the scheme of approximation of chaotic dynamics by using the GP is extended to the WMS design. In our method, each activity on WMSs is assumed to be a function having several inputs, and the combination of these functions and sequences of the functions are improved by using the GP. Links combining nodes are assumed to be flows in networks. The syntactical validity of the individuals are checked by using a kind of counter for the parse tree obtained in the prefix representation. If the validity is not satisfied, the individual is removed from the pool of individuals. In the improvement, the evaluation function is composed of the the completion of underlying objectives as well as conventional measures such as costs and waiting time. We apply the crossover operations to the individuals possessing relatively higher fitness in the reproduction phase. The mutation operation is utilized especially to determine the configuration of split operations by reintroducing some diversity. By using the result of simulation, sequences of database access obtained by the flows in WMSs are examined whether they included at least the control sequence imposed on the underlying WMS realization. The method is applied to the improved design of WMSs and the extension to the hierarchical system design.

1. Introduction

Recently, process-oriented workflow management systems (WMSs) have been widely used as a tool to facilitate business processes (BPs) [14][15]. WMSs are also available to link various subsystems of management information systems which are not necessarily located in a firm and are combined by the extranets. Designing of WMS means how to manage the flows in the systems and how to determine the capacity of nodes which are responsible to process the tasks. Current WMSs are applicable in a reliable and static manner only if the BP is assumed to be well-structured, and the specifications of WMSs are usually derived based on conventional experiences for manual transactions. Furthermore, it is assumed that WMSs are applicable in a static circumstance and there is no need for ad hoc deviations or dynamic extensions at run time.

However, it is necessary to lead the designer to improved structure of WMSs by utilizing some kind of support system rather than experience-based methods. It is also necessary to modify the system by observing the failed tasks, unplanned events and exceptions, or situations that arise from mismatches between the real processes within the organization and their computerized counterparts.

Several works demonstrated that by adding or deleting tasks as well as whole task blocks,

and changing predefined task sequences, a flexibility to support ad hoc modifications is obtainable without losing control[2][11][13]. However, the scheme for changing the configuration of WMSs is also experienced-based, and is not necessarily optimal modification.

In the paper, we propose a method for the improvement and reconfiguration of WMSs based on the Genetic Programming (GP) which becomes to know an efficient alternative for the nonlinear optimization problems [1][3][4][6]-[10]. In our method, each activity on WMSs is assumed to be a function having several inputs, and the combination of these functions and sequences of the functions are improved by using the GP [1]. In the optimization, the evaluation function is composed of the the completion of underlying objectives as well as conventional measures such as costs and waiting time.

In terms of the improvement problem included in WMSs, the Genetic Algorithm (GA) has been used to solve the routing problems for workflows by cutting or combining the routes, and the method is promising to reorganize predefined sequence of WMSs [12]. However, in the GA scheme we can not introduce any new elements or nodes to the system, and dynamic change of systems is not attainable.

In previous works, we showed the approximation of chaotic dynamics by using the GP where the primitive functions are organized by various kinds of arithmetic functions [4][5][16]. The method was successfully applied to the prediction of chaotic time series where the number of available data is relatively small. The method was also applied to the control of the chaotic dynamics generating various times series with higher dimension. In the GP, the system equations are usually represented by parse trees (called individuals as well as in the GA) [3][6]-[10]. One parse tree corresponds to a system of dynamic equations. An individual corresponds to the function which give good approximation for observed data has higher fitness. We apply the crossover operations to the individuals possessing relatively higher fitness in the reproduction phase. The mutation operation is also applied to reintroduce some diversity in an otherwise stagnant population. It is expected that the scheme of approximation of chaotic dynamics by using the GP is extended to the WMS design.

The GP used in the paper generate realizable configuration of WMSs by applying genetic operations for individuals representing WMS realizations [1]. A parse tree (individual) corresponds to a configuration of WMS in which the leaves means the input node generating documents and the nodes represent the operations to process the documents (flows). Links combining nodes are assumed to be flows in networks. The syntactical validity of the individuals are checked by using a kind of counter for the parse tree obtained in the prefix representation. If the validity is not satisfied, the individual is removed from the pool of individuals.

We also impose the conditions for sequence constraints for the database access done by the WMSs. By using the result of simulation, sequences of database access obtained by the flows in WMSs are examined whether they included at least the control sequence imposed on the underlying WMS realization.

In the crossover operations, the crosspoints are basically randomly selected, but to keep the consistency of operations on the pair of individuals to be swapped , a kind of counter is introduced to find relevant

crosspoints. To simplify the GP to the individual including split operations , we divide the population of individuals into two group, and the crossover operation is applied only one of them. The GP operations for these procedure are consistent and simple enough to apply the method to various types of WMSs.

Then, we apply the GP method to generate WMS realization for the document processing where the sequence of database access is predefined. As another applications, we

show the improvement of WMS by modifying predefined WMS by observing failed tasks and expected events. The result shows that the GP proposed in the paper provide us a comparable performance for the decision making as the conventional results, and result in the improvement of the design of WMSs.

In the followings, in Section 2, the optimization problem in WMSs design is described. Section 3 shows the algorithm for applying the GP to WMS generations. In Section 4, we show the application of the method of the paper to the design and improvement of WMSs.

2. Problems to Improve WMSs

2.1. Problem description

Current Bprocess-oriented WMSs are capable in a reliable and secure manner only if the dynamic extensions at run-time. As only few business processes are static, in this sense, this significantly limits the benefit and the applicability of current WMSs. There are several reason for the necessity of dynamic changes for workflows without losing control, for example, the dynamically evolving workflow and the specification claimed at run-time (ad hoc workflow)[2][3]. Hence the resulting requirements for the automated improvement of WMSs are far more challenging than those faced by standard transaction technology and advanced transaction models.

A basic step to improve the WMSs is the effective and efficient support of ad hoc modifications and well-aimed extensions of processes during their execution. In other words, the automated improvement must provide functions for adding of deleting tasks as well as whole task blocks and for changing predicted task sequences. Sometime, improved system allows or suggests the user to skips tasks with or without finishing them later. The system may serialize two tasks that were previously allowed to run in parallel.

But, on the other hand, we must carefully cope with the procedure of improvement of WMSs. Unrestricted changes to the structure of a long-running program may bring a kind of difficulty to have the system behave in a predictable and current manner. WF changes must not mean that the responsibility for the avoidance of consistency problems on run-time errors. The conditions claimed for the improvement of the WMSs are summarized as follows.

(1) Keeping structural dependencies between tasks.

The consistency among basic elements in WMSs such as the control, data and temporal dependency must be taken into account when the system is restructured. Otherwise, changes such as the deletion or the addition of a task may cause several inconsistencies.

(2) Cost-benefit criteria.

In the WMSs several kind of resources are involved for the execution of tasks such as the database system, computing facilities and also the human resource. Therefore, the automated improvement must show the best resolution by considering the cost-benefit criteria and several restrictions posed on the problem.

(3) Successive changes.

It is natural to understand the initial design of WMS is not so bad if the ad hoc changes are needed. In a similar manner, every modification of WMSs may suggest us the direction of improvement. Then, a kind of gradual changes and improvement based on successive changes such as the GP is useful for the underlying problems.

2.2. Tasks in WMS

WMSs are expected to coordinate and control of the activities of people and systems in an organization. Most of off-the shelf WMSs are usually defined as a software including

at least following three categories, 1)documentation software, 2)groupware management software, 3)workflow management software [14][15].

Unfortunately, the terminology used in the definition of WMS is not unique, therefore, we use following definitions which are widely recognized.

(1) documents

A statement of form used for the business transactions such as a billing statement or a travel authorization form.

(2) field

Any data element contained in a document that has a numeric or symbolic value.

(3) worker

Any person who performs operations on documents using WMSs.

(4) role

A generic identifier for a group of workers, and any one of whom may perform a task assigned to the role.

(5) work basket

A unique, logical box associated with a role, and it means a stack of documents waiting to be processed.

(6) operations

The smallest unit of work such as entering a data value onto the amount field on a form.

(7) task

A collection of operations that corresponds to a step in a common business process.

(8)sequence dependency

Two tasks that must be performed by different roles in a given sequence create a sequence dependency. For example, an invoice can be sent to cashier for payment only after manager approval.

In the following discussion, we use a simplified model denoted by a network as shown in Figure 1 consisting of nodes and links to describe WMSs for the improvement of system configuration. The operation done by a worker is represented by a node in networks. A node is characterized by several attributes such as the kind of operation and the cost of work. We define specific nodes which corresponds to generation of initial task such as a proposal of travel, and is called input nodes.

The document is relayed from node to node, and is called as a flow on networks. A node can include a buffer for flows which corresponds to the work basket. The direction of flow is represented as a link. In the context of cooperative works in WMSs, a flow corresponds to a agent who deliver and relay the information of underlying tasks. These agents are generated in the input nodes, and are combined or split in nodes until the flow (result of operations on agents) goes out of the network. The node from which the flow goes out of the WMS is called the output node.

The purpose of the design of WMS is to know the system configuration which gives minimum cost or operations within the restriction that the tasks are truly processed.

2.3. Control of flows and sequence constraints

The control of flow of documents usually includes several kinds of procedure such as restricting access to documents, read and write of documents. For simplicity, we assume we have two aspects in workflow controls to limit the operations that the workers can perform to access and modify the document. We call these operations as read/write operations. The scheme is easily extended to the status change operations such as initialize, increase, enter, decrease, abort, freeze, suspend, unsuspend the documents.

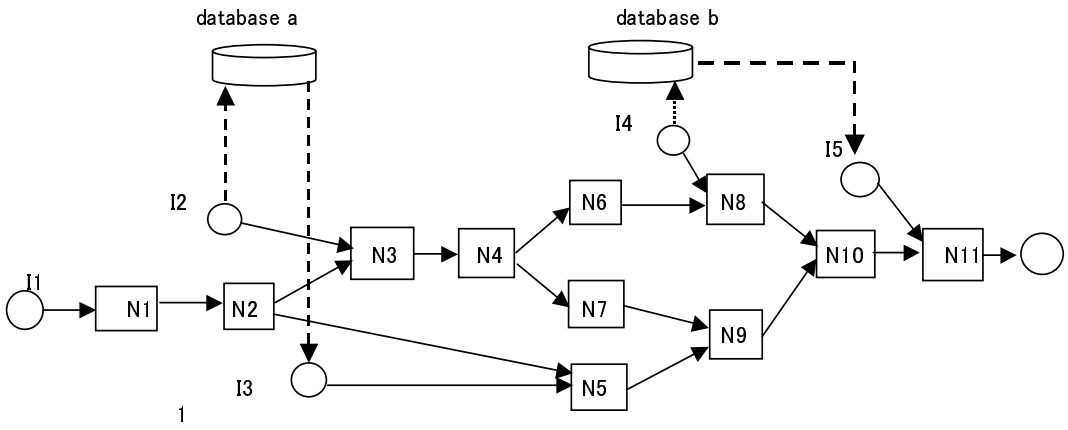

Figure 1: Schematic diagram of WMS

In addition to the read/write operations for documents, we use the approval for the final document which is usually notified by a supervisor.

In the WMSs, the sequence of these operations are represented by the paths of flows denoted as the links in the network. These paths are usually called the routing scheme. A routing scheme is the specification of routing paths that a workflow is required to follow. A routing scheme can be either mandatory where all steps must be followed in a strict prescribed order, or it may be flexible where several alternative paths are permitted so far as the order of operations and conditions for workflows are not violated.

In this paper, we are interested in flexible routing schemes. The order of operations at least the WFS must follow to realize the task is called the sequence constraint. Sequence constraints occur in many scheduling problems such as mechanical assembly, control in robotics systems and machine shop scheduling.

2.4. Operations in nodes

A routing process in nodes can be described by events, precedence (the sequential relationship between events) and operations. We assume five basic types of operations shown in Figure 2. These operations are the same as the proposal of Workflow Management Coalition [14][15].

(a) The sequential execution

The sequential execution of two tasks is modeled by connecting them with a link.

(b) The parallel processing

The modeling of branches in used to describe the parallel processing. Branches start with a split node, and from the node, the same flow is splitted to several outgoing links. We assume two types of parallel processing (split node), namely, AND-split and OR-split node. In a AND-split node, the coming flow is splitted to all of the outgoing links. In the OR-split node, the input flow is splitted to one of the outgoing link depending on a probability.

(c) The synchronization

The mechanism of synchronization of events is represented by a node with several input link and one outgoing link. We assume two types of synchronization called as a AND-join node and a OR-join node. In the AND-join node, the input flow is proceeded to the outgoing link if and only if all of the input events become ready. On the other hand, in the OR-join node, the flow is sent to the outgoing link if one of the input event is ready.

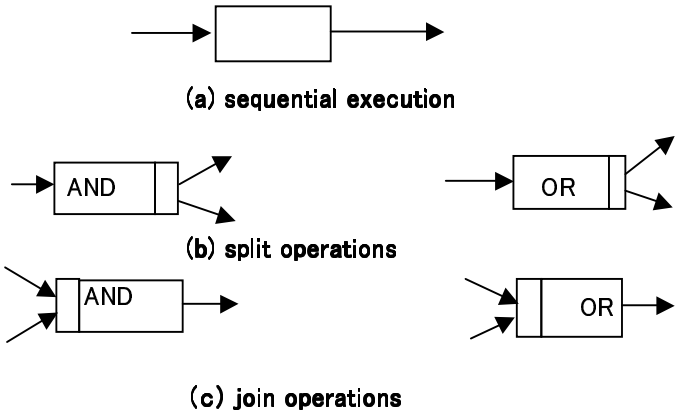

Figure 2: Representation of relation among nodes and flows

2.5. Evaluation of WMS configuration

The purpose of WMSs is the execution and completion of underlying tasks by satisfying the sequence constrains imposed on the order of flows. However, we must also note that a better configuration of WMS is obtainable by changing the attributes on nodes in such a way that we attain the cost saving and time saving by exchanging the roles and operations. In the same way, the improvement of WMS is evaluated based on a performance measure by comparing the configuration of WMS obtained before and after the application of the GP.

Then, in the following, we assume that the configuration of WMS is improved by calculating the performance of WMS based on following three points. It is also assumed that these value for evaluating the performance of WMSs are obtained by a theoretical formula or numerical simulations.

(1) waiting time of flows (W_i)

A node is regarded as a service facility which execute an operation with a certain amount of service time. Then, we obtain the total time for a document (task) which is generated in a input node, and finally delivered to the final node. We call the total time assigned to each task i as a waiting time W_i as well as the waiting time in the queuing systems.

(2) cost of node (C_i)

We also assume that a node is characterized by the cost with which the node is facilitated to perform a operation. For example, well-trained employee is expected to process the operations compared with new comers who has very little experience on the task. It is reasonable to pay more salary to the well trained employee. In the same way, we can save the processing time by using the machine with higher capacity but expensive.

(3) throughput (T_i)

The throughput of a facility is usually defined as the rate of utilization in works. If the system configuration of WMS is redundant and several nodes are not used almost always, then the throughput of the system is degraded. Then, the system configuration is seen to be irrelevant even of the waiting time and cost inherent to the configuration is relatively good.

(4) number of matches S_i in sequence constraint

WMS realization must satisfy the sequence constraint for the database access already described. Therefore, we calculate the number of matches between the sequence of access

provided by the WMS and predefined sequence by considering a kind of redundancy and the penalty if matching is not completed.

Then, the overall performance f_i of i-th realization of WMS is represented as follows.

$$f_i = k_1/W_i + k_2/C_i + k_3/T_i + k_4 S_i \tag{1}$$

where k_i are constants.

3. Improvement by Using the GP

3.1. Representation of WMSs by individuals

The GP is an extension of the conventional GA in which each individual in the population (pool of individuals) is a computer program composed of the arithmetic operations, standard mathematical expressions and variables [6]-[11]. The GP has been successfully applied to the approximation of chaotic dynamics. There are several way to represent the mathematical expressions in the GP, and among them the tree structure or symbolic expression (S-expression) of the Lisp programming language provides simple and direct representation.

We must at first design the interpreter of the genetic programming (genome interpreter) in an efficient way. The interpreter specifies the following lower level aspects of the design, namely, the representation of the nodes and the tree , the evaluation of the nodes and the whole tree, and the genetic operation on the nodes and the whole tree.

In the parse tree of a function, the node non-terminal are taken from some well-defined functions such as binomial operation $+, -, \times, /$, and the operation taking the square root of variable. Terminal nodes consist of arguments chosen from set of constant and variable. The pool of variable consists of the variable $x(t)$ itself and the time lag of $x(t)$ such as $x(t-1)$.
For example, we have the next prefix representation.

$$(6.43 \times x_1 - x_2) \times (x_3 - 3.54) \rightarrow \times - \times 6.43 x_1 x_2 - x_3 3.54 \tag{2}$$

The evaluation of prefix representation is done with the stack operation. At first, we substitute the value for variables x_1, x_2, x_3. Then, we begin to scan the prefix representation, and if we meet the terminal (operand) then we push down the term into the stack. If we meet the node (operator), then we take out the operands from the stack, and execute the operation. The result of the operation is also pushed down to the stack.

In this paper, we use the tree representation corresponds to the Lisp S-expression (called as individuals in the following) to show the configuration of WMSs. We use the terminology 'operator' and 'input flow' as the operation in WMS and the input of document at the input nodes of WMS. The relationship between the tree expression (equivalent to the S-expression) of the function and the representation of a WMS is shown in Figure 3. In this figure, leaves consist of input flows and nodes consist of operators. The links denotes the combination of nodes.

In the tree structure representing a function stand for a operator which combines the variables or intermediate value obtained by the preprocessed operations. A node in a WMS structure means the combination of events (operation), for example, a set of events which are synchronized by an AND-join node.

For example, if two flows F1 and F2 are used for the input flow of an AND-join operation, and then the result is used for a OR-join operation in the next stage with another flow F3, then we have the prefix representation for the WMS as

```
OR-join AND-join F1 F2 F3
```

In the prefix representation, a operation is followed by two flows or the result of operation. The result of operation in a node is used further as a tentative (intermediate) flow in the next stage of operation. Then, an individual is interpreted as follows. We begin to scan the string of an individual to find a combination of a operation and two flows. Then, we execute the operation and the result is stored in a register in the interpreting system as an intermediate flow. By successive execution of operations in the same way, we have finally a single flow which corresponds to the result of output node. In the interpretation mentioned above, we also calculate the performance measure such as the waiting time for each flow.

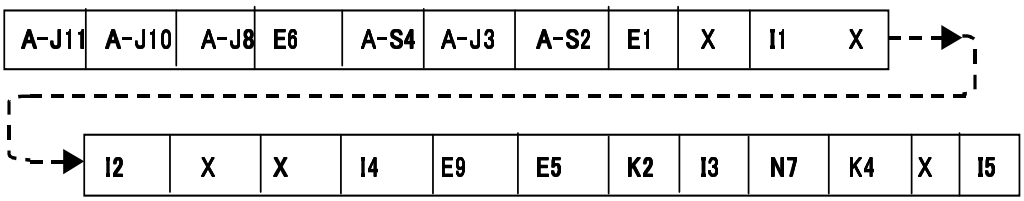

Figure 3: An example of individual representing WMS

As a result, we find that a WMS is able to be represented by a tree structure which is usually used to represent functions. We apply the crossover operations to the individuals possessing relatively higher fitness in the reproduction phase. The mutation operation is also applied to reintroduce some diversity in an otherwise stagnant population.

3.2. Introducing dummy input flows

In the operation of sequential execution, the number of coming flow and outgoing flow are one. As already mentioned, we restricted ourselves to the case where the input flows for a operation is two. It is not relevant to treat the operation of sequential execution as an exception. Therefore, we introduce a kind of dummy node of input flow for the operation of sequential execution. The node acts as a synchronization which is always ready.

3.3. Representation of split

In the operation of AND-split and OR-split, the result of operation which is also a tentative flow is used for a input for another operation which is not necessarily adjacent to the underlying operation. In principle, the interpretation of such splits is not difficult and managed by a table consisting of a source and destination of splits. However, in the GP operation discussed later, the crossover for two individuals including splits is very hard in maintaining the consistency. Moreover, a specific procedure for treating the crossover of splits make the GP system very complicated.

Therefore, we employ a simple two stage scheme to process the splits in the GP. It is assumed that the crossover operation is applied only to the 'basic' form of individuals, and the individual with completed split operations are generated by using the mutation operations. Figure 4 shows the overview of the method.

At first, we generate a population of individuals (Population 1) which include split operations as well as join operations. However, the destination to which the result of split operation is delivered is not determined in the stage (called first stage). The destination of the operation is determined in the second stage. Namely, we select at random possible flows which are able to be regarded as the destination in an individual, and then determine

one of them as the destination of the split operation. This procedure corresponds to the mutation operation in the conventional GP.

Then, we have a population of individuals (Population 2) obtained by applying the mutation operations to individuals in Population 1. We generate two offsprings in Population 2 for each individual in Population 1. After obtaining the individual representing WMSs , we evaluate and the fitness of individuals in Population 2. The fitness of individuals in Population 2 is reflected to the fitness of Population 1 from which the offsprings are generated.

The improvement of fitness of individuals by the GP is done by applying the crossover operation only to Population 1. Then we apply the mutation operation to the new population obtained by the crossover operation, and then we have new set of individuals (offsprings). The individuals having lower fitness in Population 2 are replaced by these newly generated offsprings having relatively high fitness.

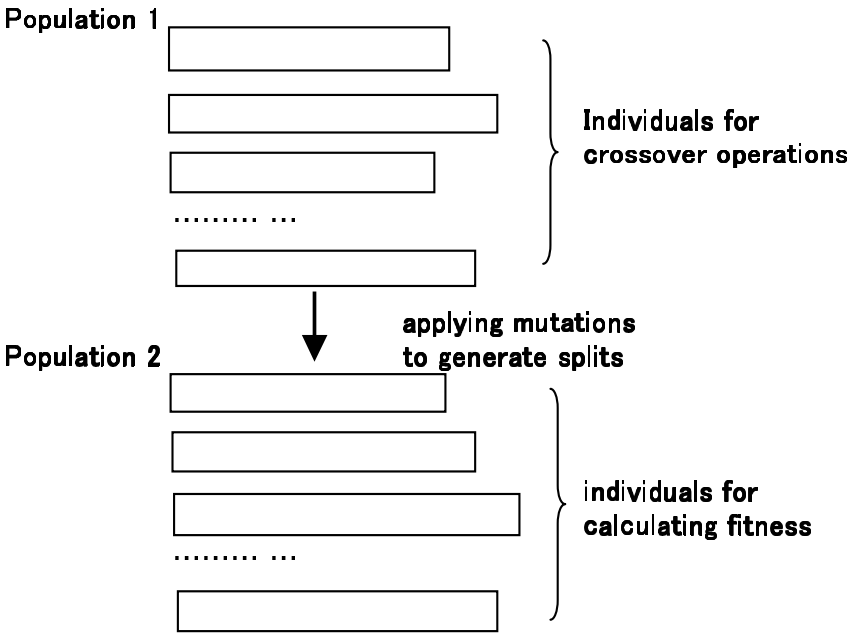

Figure 4: Two stage GP operations

3.4. Initial values and consistency

In the GP, the initial values of individuals are assigned by using the random numbers. However, the content of each individual is needed to be syntactically valid, and also the result of genetic operation is necessary to be relevant to represent the parse tree.

For checking the validity of underlying parse tree, the so-called stack count (denoted as *StcckCount* in the paper) is useful [6]. The *StackCount* is the number of arguments it places on minus the number of arguments it takes off from the stack. The cumulative *StackCount* never becomes positive until we reach the end at which point the overall sum still needs to be 1.

By using the *StackCount* we can see which loci on the prefix expression is available to cut off the tree for the crossover operation, and we can validate whether the mutation operation is allowed.

If final count is 1, then the prefix representation (tree) corresponds properly to a system

equation. Otherwise, the tree structure is not relevant to represent the equation.

If we apply the *StackCount* to the parse tree of WMSs, we regard the operators and the operands in the representation of functions as the operations and flows in the WMS representation, respectively.

Additionally, the individuals corresponding to WMSs must satisfy the workflow sequence constraints which arise from the routing scheme , i.e., the routing paths that the workflow is required to follow. For example, we impose the constraints on each flows that flows must visit the database a (operation is 'read'), then the database b (write), and finally the database c (read) with obtaining approval from the supervisor. If the individual representing WMS realization do not satisfy the constraint, then the individual is not included in the pool.

Several theoretical tools are available to check the validity of individuals, but we use a simple method based on the simulation of WMSs. As already mentioned, the performance of WMSs is evaluated by observing the characteristics of WMS such as the waiting time of flows. Therefore, the sequences of database access of each flow is also observed and recorded in the simulation. The overview of the method is shown in Figure 3. In this case we must note that in the join node, the sequence of database access of two input flows is merged and succeeded to the outgoing flow. At the same time, in the split node, the sequence included in input flow is succeeded to several nodes which are the destination of split operation.

Then, we have a set of sequences representing the database access of flow corresponding to the underlying individual. It is assumed that the sequences can contain some kind of redundant expressions for the database access. For example, if the sequence P=(a,b) means the sequential access to the database a and b, we also accept the sequential access Q=(a,a,b) which included duplicated access to the database b.

By generating all possible redundant patterns of the sequences, we can examine whether the system can satisfy the sequence constraint.

3.5. Crossover operation

Contrary to the operation in the GA, the crossover operation in the GP is applied to restricted cases. To keep the crossover operation always producing syntactically and semantically valid programs, we look for the location on individual which can be a subtree in the crossover operation and check for no violation. By using the *StackCount* already mentioned, we know the subtrees which are the candidate for the crossover operation. The basic rule is that any two loci on the two parents genomes can serve as crossover points as long as the ongoing *StackCount* just before those points is the same. We show an example in Figure 4.

At first, we select a random position in individual A and calculate the *StackCount* up to this location (for example N_a). Then, search corresponding crosspoints in individual B, as long as the *StackCount* on these crosspoints is the same as N_a of individual A. Then, we decide one crosspoint in individual B at the same probability. The crossover operation creates new offsprings by exchanging sub-trees between two parents.

3.6. Mutation

The goal of the mutation operation is the reintroduction of some diversity in an population. Two types of mutation operation in GP is utilized to replace a part of the tree by another element.

(Global mutation :G-mutation)

Generate a individual I_s, and select a subtree which satisfies the consistency of prefix

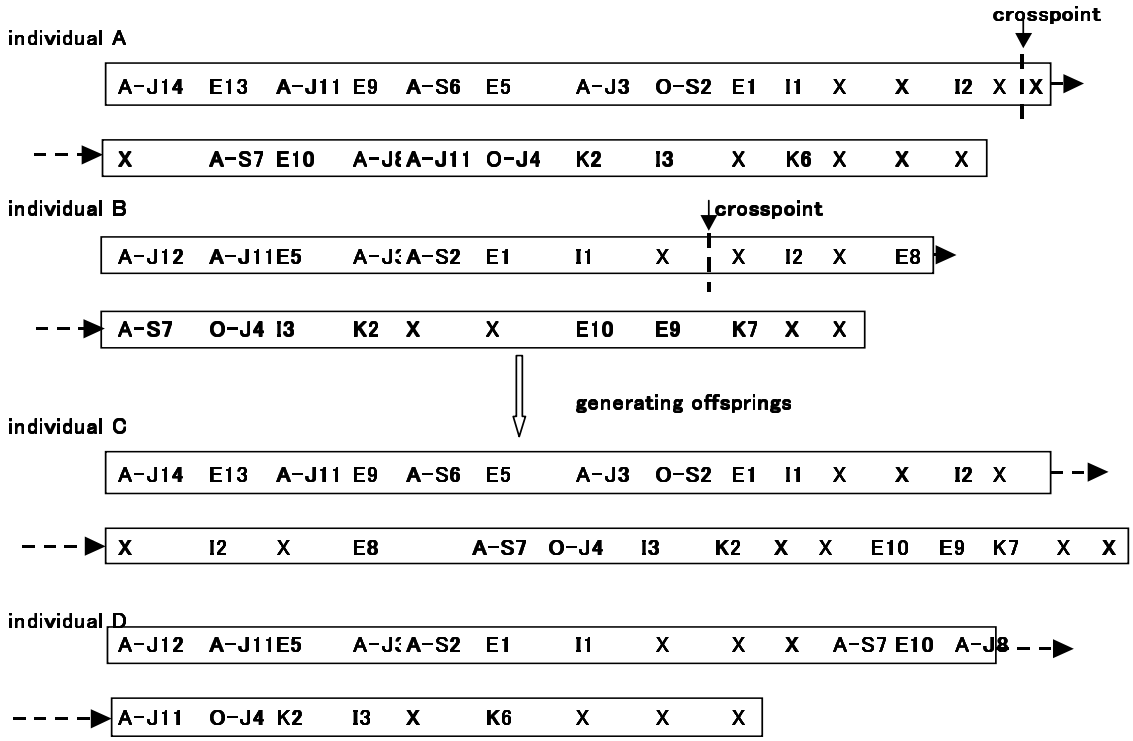

Figure 5: Crossover operation

representation. Then, select at random a terminal in the individual, and replace the terminal by the subtree of the individual I_s.
(Local mutation:L-mutation)

Select at random a locus in a parse tree to which the mutation is applied, we replace the place by another value (a primitive function or a variable).

3.7. Improvement by using the GP

By using the measure of fitness to evaluate each individual, we apply the GP to the population to derive better configuration of WMS.

We iteratively perform the following steps until the termination criterion has been satisfied.
(Step 1)

Generate an initial population of random composition of possible functions and terminals for the problem at hand. The random tree must be syntactically correct program.
(Step 2)

Execute each program (evaluation of WMS) in population and assign it a fitness value giving partial credit for getting close to the correct output. Then, sort the individuals according to the fitness S_i.
(Step 3)

Select a pair of individuals chosen with a probability p_i based on the fitness. The probability p_i is defined for ith individual as follows.

$$p_i = (S_i - S_{min}) / \sum_{i=1}^{N} (S_i - S_{min}) \tag{4}$$

where S_{min} is the minimum value of S_i, and N is the population size.

Then, create new individuals (offsprings) from the selected pair by genetically recombining randomly chosen parts of two existing individuals using the crossover operation applied at a randomly chosen crossover point. The algorithm is the same as the roulette strategy. If the individual having highest fitness is not included, then we apply the strategy of elite preservation. Iterate the procedure several times to replace individuals with lower fitness. (Step 5)

If the result designation is obtained by the GP (the maximum value of the fitness become larger than the prescribed value), then terminate the algorithm, otherwise go to Step 2.

3.8. On the feedback loop

It is natural to consider the case where the execution of tasks or processing done to the document is irrelevant. In this case the documents are re-transmitted to the preceding nodes for the amendment. In the configuration of the networks of nodes, the transaction stated above corresponds to the feedback loop.

Unfortunately, in our paper we do not consider the function and the meaning of feedback loop for processing the document. In the GP procedure, we search for the better individual and configuration of nodes which can show us better combination and connection of nodes under the limitation the document must visit each database in a predefined sequence (sequence dependency). If any one flow of these possible flows satisfy the sequence dependency, we regard that the system is acceptable even though several documents are re-transmitted through the feedback loops, and are not included in the evaluation of the performance (fitness) of the individuals.

The existence of the feedback loop is easily detected by observing the connection matrix (incident matrix) of the nodes which is generated based on the information of nodes embedded in the strings of individuals.

Thus, the feedback loop is not specifically considered in this section.

However, the retransmission of document and the design of feedback loop are important tasks for WMSs, then we describe further works necessary to improve the WMSs design in the Discussion.

4. Applications

4.1. Authorization of documents

The GP method of the paper is at first applied to the design of WMS for the authorization of documents in four cases. We assume there are three database a,b and c, and the sequence constraints for cases are as follows.

(case 1) read a, read b , read c

(case 2) read a, read/write b, write c

(case 3) read /write a, read/write b, write c

We also assume the processing time for nodes and the arrival rate of the input nodes are given as follows.

processing time :normal distribution $\mu = 2.0, \sigma^2 = 0.02$

arrival rate:Poisson distribution with $\mu = 10$

The parameters in equation (1) are as follows. $k_1 = k_2 = k_3 = 1, k_4 = 8$

The coefficient k_4 is relatively large, while the sequence constraint for the WMS is substantial at the begining of optimization. The cost of a WMS realization C_i in equation (1) is proportional to the value defined as the number of operation nodes N_n and input nodes N_i. But, so as to suppress the complexity of WMS configuration, the surplus of N_n and N_i,

namely, $\max(1, N_n - N_1)$ and $\max(1, N_i)$ is used for the calculation where N_1 and N_2 are assumed to be minimum number of nodes necessary to realize the system.

In terms of the satisfaction of sequence constraint S_i in equation evaluated as the number of matches of sequence L_1 generated by the WMS realization compared to the predefined sequence L_2. For example, if $L_1 = (a, b, c)$ and $L_2 = (a, b, c)$ then the score is three. If $L_1 = (a, b, b)$ and $L_2 = (a, b, c)$ then we impose the penalty one since the matching of sequence is terminated at the second symbol, and it remains one symbol unused.

The parameters of the simulation for the GP are selected as follows.
Population size=500
Maximum size of array=25
Maximum generation of operation=500
Probability of mutation (in Population 1)=0.01

Figure 6 shows an example of ultimately obtained configuration of WMS. Figure 7 shows an example of the relation between the fitness and the generation of the GP. The result of improvement is shown in Table 1. In Table 1, N_1 and N_2 mean the minimum number of nodes for the realization mentioned above, and f_0 and f_T mean the initial and final value of the fitness which is the largest in population. Table 1 also shows the number of iterations denoted as N_p at which the point we have almost the improved configuration of WMS as the ultimately obtainable results.

As is seen from the result, the fitness of the population is increased by applying the GP, and it results in the improvement and automatization of the WMS design by the GP. In general, we have almost improved configuration of the WMS at about 100 GP operations.

Even though the satisfaction of sequence constraint is embedded in the evaluation function (fitness), the final result shows the improved configuration satisfying the constraints.

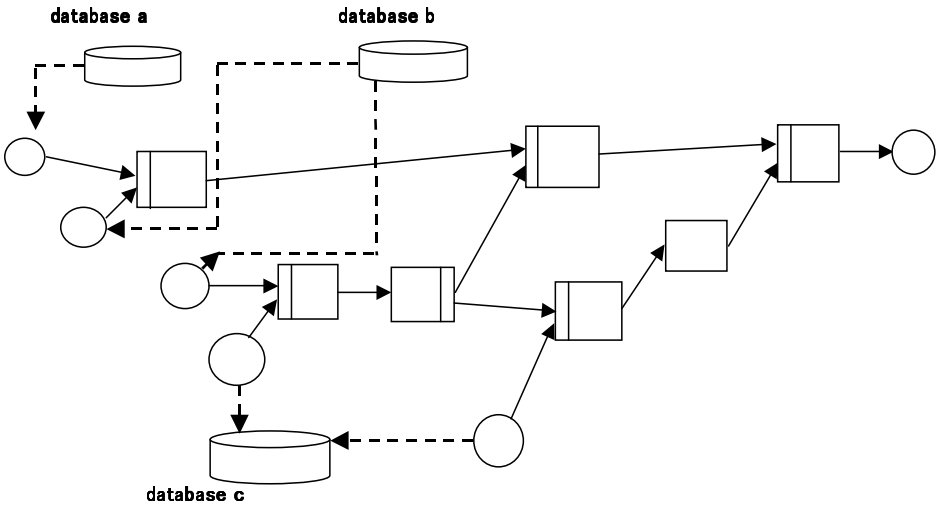

Figure 6: An example of improved WMS configuration

4.2. Effect of restriction on activities in nodes

In previous section, we assumed the processing time in nodes are determined by a random variable, and as a result, the restriction on the human resource represented by the worker who are assign to roles is no considered. Generally speaking, the budget allowed to run the WMSs is restricted as well as the restriction on the properties of workers. Therefore, it is

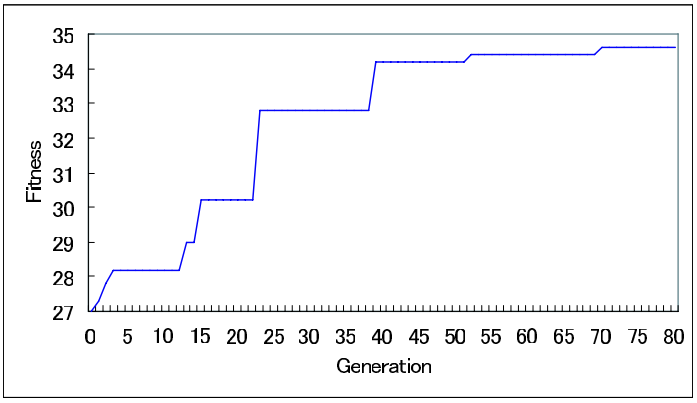

Figure 7: Relation between the fitness and the GP generations.

Table 1: Result of improvement

case	N_1	N_2	f_0	f_T	N_p
case 1	5	5	17.2	35.6	67
case 2	8	6	17.2	34.5	121
case 3	10	8	17.1	33.5	180

necessary to analyze the cases where the activities in nodes are restricted.

In this section, we assume that the property of nodes is determined by searching a kind of look-up table, and sometime we fail to assign good worker who is responsible for a role in nodes. The scheme corresponds to the case where the budget and the human resource is restricted, and the assignment of the worker may affect the improvement of WMSs.

However, the comprehensive simulation study for these cases may changes the basic purpose of our paper, then, in the following, we simplify the problem as a selection and assignment problem. We have four kinds of nodes, namely, AND-join, OR-join, AND-split and OR-split. For example, for the AND-join ,we assume that for each node, we can assign on of the workers denoted as $a_1, a_2, ..., a_M$ where M (called as a feasible set A of allocation) is the maximum number of available worker. In a similar manner, we define feasible sets B, C, and D for OR-join, AND-split and OR-split.

As the properties of workers, we assume the processing time (abilities) of workers and the cost (wages) of workers. In addition to these properties, we assume also the reliability of workers. In general, if the wage of a worker is relatively high, then the worker is reliable. We find an appropriate worker from a look-up tables. In a similar manner, we make another three look-up tables for OR-join, AND-split and OR-split nodes.

We assume also that the assignment of a worker to a node is done at a random manner. In case if the management system fails to assign appropriate workers to nodes, and the shortage of worker is found, then the penalty is added to the evaluation of the individual in the GP process. We call the case as the allocation of worst-case worker a_w. It is also assumed that in the offsprings generated after the GP operation, if the individual includes the same worker in different nodes, then we replace one of them by worst-case worker.

Table 2 (called Scheme I) and Table 3 (called Scheme II) show the list of properties in the simulation study. Scheme I corresponds to the case with relatively small restriction of the design. In contrast to Scheme I, in Scheme II available resource is very restrictive, and it is relatively hard to apply the improvement. Another conditions for the simulation are the same as the study in Section 4.1.

In Tabel 2, the number of worker is relatively large compared to the case represented in Table 3. Table 3 corresponds to the case where the number of available workers is restrictive, and the system needs ah doc allocation of workers.

Table 2: Propeties of workers (Scheme I)

	a_1	a_2	a_3	a_4	a_5	a_w	b_1	b_2	b_3	b_4	b_5	b_w
time	1	2	1	3	4	5	7	6	7	8	5	10
wage	3	1	3	2	1	2	5	7	5	4	8	3
reliability	10	5	10	7	5	6	4	7	4	3	10	3
	c_1	c_2	c_3	c_4	c_5	c_w	d_1	d_2	d_3	d_4	d_5	d_w
time	3	4	3	5	7	8	8	6	5	6	7	8
wage	3	2	3	1	1	2	3	2	5	4	3	2
reliability	10	7	10	4	4	2	3	2	10	9	5	2

Table 3: Properties of workers (Scheme II)

	a_1	a_2	a_w	b_1	b_2	b_w
time	2	1	5	8	5	10
wage	2	3	2	4	8	3
reliability	7	10	6	3	10	3
	c_1	c_2	c_w	d_1	d_2	d_w
time	3	5	5	3	4	8
wage	6	4	2	6	5	3
reliability	6	4	2	10	6	2

To include the evaluation for the reliability, the fitness f_i defined in equation (1) is extended. Namely, we add a term $d_5 R_i$ to the fitness for ith individual where R_i is the total value of reliability of the system realized by the string of ith individual. In the system for ith individual, for the serial connection of nodes the reliability is simply added. For the case of parallel processing, the highest value of the reliability among the nodes is regarded as a value of reliability. In the following, we set $k_5 = 0.2$.

Table 4 and 5 summarize the result of simulation where the symbols such as N_1, N_2 have the same meaning as in Section 4.1. As is seen from the result, the generation of the GP procedure are relatively large compared to the cases in Section 4.1. This means that the restriction on the budget and the number of properties make the improvement of the WMSs a little hard and it becomes time consuming, but , the improvement of the WMSs based on the GP is reliable.

Table 4: Result of improvement(Scheme I)

case	N_1	N_2	f_0	f_T	N_p
case 1	5	5	37.8	57.6	150
case 2	8	6	37.2	54.9	181
case 3	10	8	37.4	53.5	220

Table 5: Result of improvement(Scheme II)

case	N_1	N_2	f_0	f_T	N_p
case 1	5	5	47.2	55.6	267
case 2	8	6	47.9	54.5	281
case 3	10	8	47.1	53.5	380

4.3. Extension to hierarchical WMS design

Now we extend the method of the paper to the hierarchical design of WMSs where the number of tasks are relatively large. Recently, the web-service systems in the Electronic Commerce (EC) is introduced for the purpose to automatize the procurement of material necessary for the production. Programs on the web-service system such as the UDDI (Universal Description, Discovery and Integration) are able to be regarded as the agents who search and inform the producer the relevant materials for production. The improvement on of WMS realization is extended to the design of agents in web-service systems.

However, it is imagined that the number of tasks included in the web-service is relatively large, and the direct application of the GP method for the flow control is not reliable. Then, we divide the whole system into several subsystems each of which is designed based on the method treated in the paper.

If the policy to divide the whole system into subsystems is determined, the way to apply the GP method for the improvement of the system is not so complicated. For example, if the tasks for a subsystem S_1 is composed of a, b, and c (the set of tasks is denoted as A), and then the completion of these tasks initiate the another set of tasks composed c, d and e (denoted as B) in the subsystem S_2 as shown in Figure 8. Then, the newly defined sets A and B can be regarded as the tasks in the original scheme of the improvement of WMS design, for example, by replacing the task a by the task A.

Table 2 shows an example for discussing the effect of introducing the idea of subsystems.
(case 1) number of tasks=7, division=2
(case 2) number of tasks=10, division=3
(case 3) number of tasks=12, division=4

In Table 2 we also show the number of GP operations (denoted as N_{GP}) necessary for the improvement of the WMS without dividing the whole system. It is easily read from the result, if we do not divide the whole system, the GP procedure is not more applicable if the number of tasks increases over 10.

Table 6: An example for reduction of computation

case	N_{GP} without division	N_{GP} with division
case 1	2200	302
case 2	3123	512
case 3	-	634

As discussed above, the division of whole system into several subsystems is effective in the design of large sized system. However, it may occur the cases where the tasks in a subsystem A affect the execution of tasks in another subsystem B. Then the design of a WMS in a subsystem is irrelevant if we neglect the influence of these kind of tasks.

The problem is treated by introducing tentative database x as shown in Figure 9 which

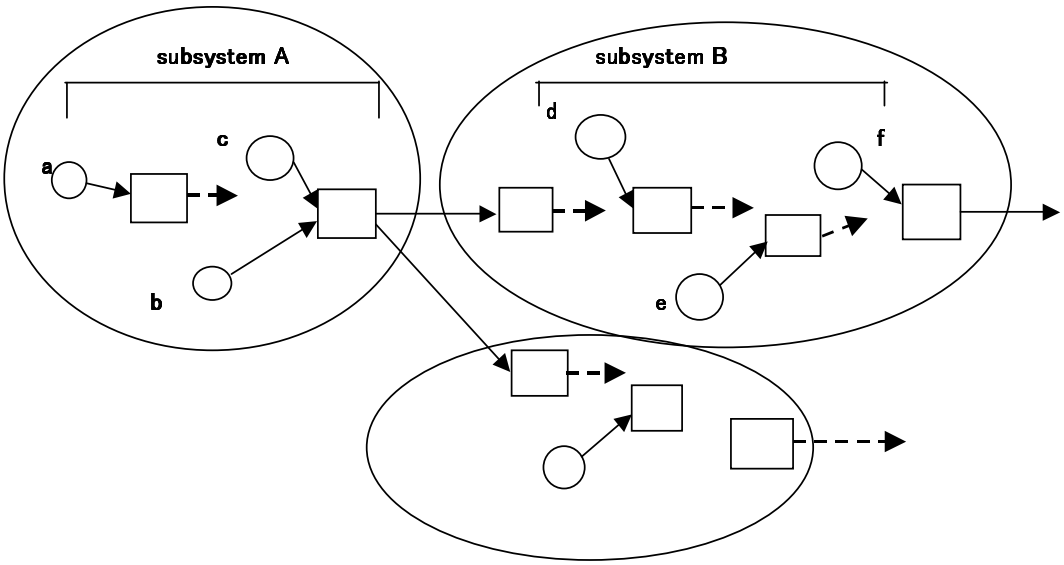

Figure 8: An example of subsystems

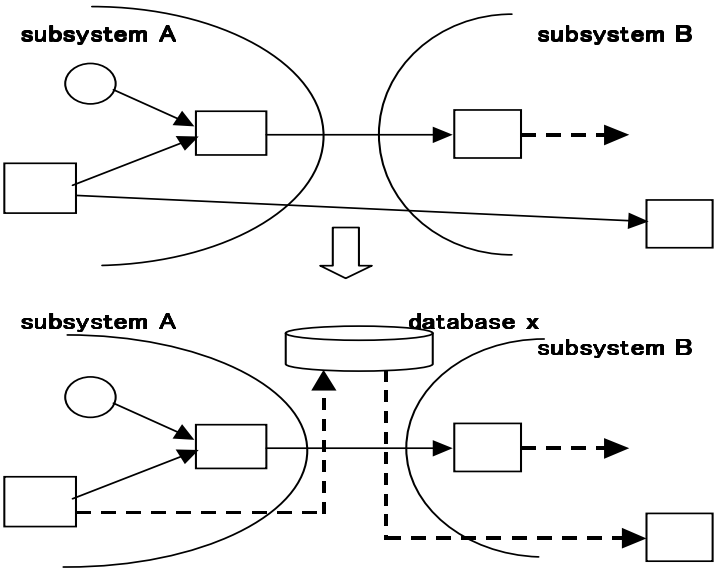

Figure 9: Processing the task ranging over two subsystems

Table 7: Result of improvement

case	N_{GP} with division
case 1	502
case 2	612
case 3	734

transmit the completion of tasks in subsystem A to the subsystem B and then the information is used to initiate another task. This procedure may correspond to the transmission of partial information among subsystems in real world.

Then, we evaluate the method by using the simulation. We assume that the whole system includes several places where these kind of tentative database is introduced and the number is denoted as R. We consider following three cases as discussed in the example in Table 2 by gradually changing the complexity of the problem.
(case 1) number of tasks=7, division=2, $R = 2$
(case 2) number of tasks=10, division=3, $R = 3$
(case 3) number of tasks=12, division=4, $R = 4$

Table 7 show the result of improvement for these cases. As is seen from the result, the introduction of tentative database increase the number of GP operations necessary to obtain the final result, but the convergence to the improved design is attainable.

5. Discussion

5.1. Addition and evaluation of feedback loops

In this paper, we do not treat the problem of retransmission of document for the amendment back to the preceding nodes. The scheme corresponds to the feedback loop in the network. In real applications, we must copy with the problem on the basis of GP approach. For the extension of the GP approach, it is natural to include the probabilistic event which represents a kind of irrelevant processing in nodes. Then, the document must be retransmitted to a preceding node for the amendment. If the individual has no feedback loop for the retransmission, then the fitness of the individual is decreased as a penalty.

In this manner, we can extend the GP approach to the case of feedback loop. However, it is also a kind of quantitative analysis of WMSs in the same framework of this paper, and is not capable of testing the consistency of amendment. To evaluate the feedback, we must know the error of processing and the meaning and necessity of processing on nodes in a qualitative sense. For these studies, we must use quite another approach to the design of WMSs, and it goes over the scope of problem description of this paper.

5.2. Extension to the design of another systems

In this paper, we showed another application of the GP rather than usual approximation of functions for the chaotic dynamics and finding the production rules based on the observation. In our approach, the configuration of networks is represented as a string of individual, and the GP operation for the individuals give us a way to find better configuration.

It is expected to extend the GP approach to improve the configuration of networks such as the IP networks and the routing scheme for packet transmission. In conventional analyses, the Genetic Algorithm (GA) is applied to find a alternative path for routing by cutting and joining the network by using a random selection of nodes. However, the approach is not systematic. The GP procedure may provide more relevant method for finding alternative path for routing.

In a similar manner, the GP may useful to find the reallocation of resources in the large systems such as the production systems. However, the problem for each application is not uniform and sometime it includes problem-dependent tasks to be solved, and the direct implication of GP may not bring successful result.

6. Conclusion

This paper showed an improvement method for WMSs based on the reallocation of flows using the GP and its applications. In our method, each activity on WMSs was assumed to be a function having several inputs, and the combination of these functions and sequences of the functions were optimized by using the GP. To simplify the method two stage of GP for two kinds of population were employed. In the prefix representation, if the validity is not satisfied, the individual was removed from the pool of individuals. Sequences of database access obtained by the flows in WMSs were examined whether they included at least the control sequence imposed on the underlying WMS realization. The result showed that the GP proposed in the paper provide us a comparable performance for the decision.

The problems remained to be solved include the extension of the method to various types of dataflows such as the Web-service and efficient design method. Further works will be continuously done by the authors.

References

[1] X. Chen and S. Tokinaga: A method of multi-objective optimization for the flow-type jobs using the Genetic Programming (in Japanese). *Technical Report of IEICE*, NLP2000-140 (2001) 1-8.

[2] C. Ellis, et.al.: Dynamic change within workflow systems. In *Proceedings of Conference on Organizational Computing Systems (COOCS'95) ACM* (1995) 10-21.

[3] H. Iba: *Genetic Programming (in Japanese)* (Denki-dai Shuppannkai, 1996).

[4] Y. Ikeda and S. Tokinaga: Approximation of chaotic dynamics by using smaller number of data based upon the genetic programming and its applications, *IEICE Transaction.*, **E83-A-8** (2000) 1599-1607.

[5] Y. Ikeda and S. Tokinaga: Controlling the chaotic dynamics by using approximated system equations obtained by the genetic programming. *IEICE Transaction.*, **E84-A-9** (2001) 1888-2000.

[6] M. J. Keith and M. C. Martin: Genetic programming in C++: Implementation issues. In (ed) K. E. Kinnerar, Jr. ,*Advance in Genetic Programming* (MIT Press, 1994).

[7] J. R. Koza: *Genatic Programming* (MIT Press, 1992).

[8] J. Koza: Genetic programming:A paradigm for genetically breeding populations of computer programs to solve problems. *Report No.STAN-CS-90-1314*, (Dept.of.Computer Science Stanford University, 1990).

[9] J. Koza: Evalution and subsumption using genetic programming. In *Proceedings of the First European Conference on Artificial Life* (MIT Press, 1991) 110–119.

[10] J. Koza and J. P. Rice: Automatic programming of robots by using genetic programming. In *Proceedings of 10th AAAI* (1992) 194–201.

[11] A. Kumar and J. L. Zhao: Dynamic routing and operational controls in workflow management systems. *Management Science*, **45-2** (1999) 253-272.

[12] K. Okumura, T. Tanaka and H. Ishii: Adaptive routing and flow control by genetic

algorithm with flow model (in Japanese). *IEICE Transaction.*, **J84-D-I-11** (2001) 1563-1572.

[13] M. Reichert: ADEPT:flex-supporting dynamic change of workflow without losing control in workflow management systems. *Journal of Intelligent Information Systems*, **10** (1998) 93-129.

[14] Workflow Management Coalition: The workflow reference model. *Document No. WfMC-TC-1006* (1994).

[15] Workflow Management Coalition: Terminology and Glossary. *Document No. WfMC-TC-1011* (1996).

[16] M. Yakabe and S. Tokinaga: Applying the genetic programming to modeling of diffusion processes by using the CNN and its applications to the synchronization (in Japanese). *IEICE Transaction.*, **J85-A-5** (2002) 3061-3073.

Shozo Tokinaga
Graduate School of Economics
Kyushu University
6-19-1 Hakozaki, Higashi-ku
Fukuoka 812-8581, JAPAN
E-mail:
tokinaga@en.kyushu-u.ac.jp