Vol. 42 No. 2 IPSJ Journal Feb. 2001

Regular Paper

An Integrated CAC and Routing Strategy for High-speed Large-scale Networks Using Cooperative Agents

Leonard Barolli,[†] Akio Koyama,[††] Takako Yamada[†]
and Shoichi Yokoyama[†††]

The routing algorithms can be classified into source, distributed and hierarchical routing. Source routing algorithms are conceptually simple, but they suffer from scalability problem. Distributed routing algorithms are more scalable, but loops may occur, which make the routing to fail. Hierarchical routing has been used to cope with the scalability problems of source routing in large internetworks. The hierarchical routing retains many advantages of source routing. It also has some advantages of distributed routing because the routing computation is shared by many nodes. To cope with high-speed networks, the traffic control methods must be adaptive, flexible, and intelligent. Use of intelligent algorithms based on fuzzy logic, genetic algorithms and neural networks can prove to be efficient for traffic control in high-speed networks. In this paper, we propose an integrated CAC and routing strategy using cooperative agents. The proposed routing algorithm is a combination of source and distributed routing. It uses source routing inside a domain and hop-by-hop routing for inter-domain. The proposed strategy is able to avoid flooding and routing loops, reduce the search space, and can be easily scaled-up to cope with large-scale networks.

1. Introduction

The routing strategies can be classified into three classes: source, distributed and hierarchical routing[1)~6)]. Source routing algorithms are conceptually simple, easy to implement, debug, evaluate, upgrade, and can guarantee loop-free routes. But, in source routing algorithms, the global state maintained at every node has to be updated frequently to cope with dynamic changes in the network. This result in a high communication overhead for large-scale networks. Also due to the imprecision of the global state the routing algorithm may fail to find a feasible path[1)~3)].

Distributed routing algorithms are more scalable and the routing response time can be made shorter. But, it is difficult to design efficient distributed algorithms for NP-complete routing problems, because there is no detailed topology and link state information available. Also, in the distributed routing algorithms loops may occur, which make the routing to fail[2)~4)].

Hierarchical routing has been used to cope with the scalability problems of source routing in large internetworks. Hierarchical routing scales well because each node only maintains a partial global state where groups of nodes are aggregated into logical nodes. The hierarchical routing retains many advantages of source routing and has also some advantages of distributed routing. But, in the conventional hierarchical routing[5),6)], because the network state is aggregate additional, some imprecision is introduced, which has a significant negative impact on Quality of Service (QoS) routing. Furthermore, the selection of peer group leader is difficult.

Routing algorithms for high-speed networks should have a fast decision and the routing decision should be made at source node in order to avoid computations at intermediate nodes. They should be distributed for purposes of reliability, have high throughputs, be scalable, be adaptive to network changes, and avoid the information flooding. Therefore, to cope with high-speed networks, traffic control methods must be adaptive, flexible and intelligent[10)]. Use of intelligent algorithms based on Fuzzy Logic (FL), Genetic Algorithms (GA) and Neural Networks (NN) can prove to be efficient for high-speed networks[7)~11)].

In this paper, we propose an integrated Call Admission Control (CAC) and routing strategy using cooperative agents. The proposed strategy is based on Distributed Artificial Intelligence (DAI) approach, which deals with design

† Faculty of Literature and Social Sciences, Yamagata University
†† Faculty of Computer Science and Engineering, The University of Aizu
††† Faculty of Engineering, Yamagata University

of artificial agents to develop intelligent systems. We introduce two types of agents: simple and intelligent agents. The intelligent agents are based on FL and GA. After a CAC agent has decided to accept a connection in the network, a routing agent is activated to find a feasible path. The proposed routing algorithm is a combination of source and distributed routing. It uses source routing inside a domain and hop-by-hop routing for inter-domain. The proposed strategy is able to avoid flooding and routing loops, reduce the search space, and can be easily scaled-up to cope with large-scale networks.

The paper is organized as follows. Next Section gives a brief introduction of DAI approach. The proposed distributed network architecture is treated in Section 3. The Resource Management Agent (RMA) is introduced in Section 4. The Precomputation Agent (PA), which includes Search Space Reduction Agent (SSRA) and Tree Model Network Agent (TMNA), and Destination Discovery Agent (DDA) are treated in Section 5. The Routing Agent (RA) with its Intra Domain (IntraD) and Inter Domain (InterD) agents is discussed in Section 6. Some simulation results are discussed in Section 7. Future work is introduced in Section 8. Finally, conclusions are given in Section 9.

2. DAI Approach

The high-speed networks will have to manage an increasing usage demand, provide support for a significant number of services, guarantee their QoS, and optimize the utilization of network resources. The control in these networks becomes very complex and it seems imperative to focus on a new control perception that introduces intelligence, which can enable the network to perform adaptive behavior and to decompose the control to handle complexity while ensuring cooperation between different elements of control.

The term intelligence we use in the sense of control elements which have reasoning capacities, exhibit behavioral autonomy, and are able to interact and cooperate to achieve collective work. This is related to DAI which deals with design of artificial agents to develop intelligent systems [12]. One of most important subfield of DAI is Multi-Agent System (MAS) paradigm, based on the idea that simple or complex activities are the outcome of interaction between relatively independent entities called **agents**. A MAS may then be defined as a set of agents that interact with each other and with the environment to solve a particular problem.

The term agent generally is defined as a physical or logical entity that has the following properties:

Social Ability—An agent is able to communicate with other agents. The agents may work toward a single global goal or separate individual goals.

Autonomy—Agents operate without the intervention of other agents. They can accept, or not, requests coming from other agents and have some kind of control over their actions and internal states.

Reactivity—Agents perceive their environment and respond in a timely fashion to change that may occur in it.

Adaptability—Agents are characterized by their flexibility, adaptation, and facility to set up their own goals based on their interests.

Granularity Degrees—Agents may have different degrees of complexity. They may be simple or complex. Simple agents are characterized by the lack of intelligence. More complex agents are called cognitive or intelligent agents.

3. Proposed Distributed Network Architecture

The proposed network architecture is a MAS. The agents are distributed and cooperate together. Each Domain Management Agent (DMA) has four agents: RMA, DDA, PA and RA. The DMA structure is shown in **Fig. 1**. The PA includes SSRA and TMNA. We call these two agents PA, because they make the computation before the RA is activated. The computation time starts when a new connection makes a request to the network. The RA has the IntraD and InterD agents. In fact, the InterD agent is a composition of IntraD agent and Connectivity Management Agent (CMA), which are activated by an escalation strategy. The distributed network architecture with DMAs is shown in **Fig. 2**. This architecture can be considered as a hierarchical architec-

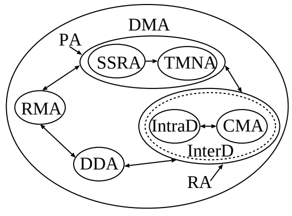

Fig. 1 DMA structure.

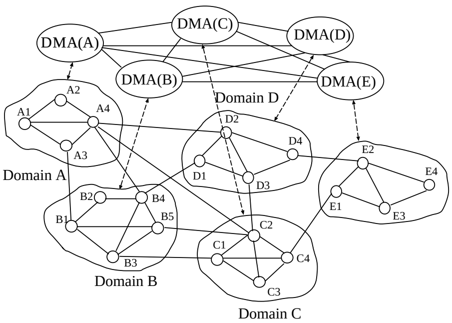

Fig. 2 Distributed network architecture with DMAs.

ture, where in first level are domains and in the second level are DMAs. We have shown here only five domains. But, this architecture can be scaled-up easily by increasing the number of DMAs and domains in order to deal with the increasing users demands and number of switches.

4. RMA

The RMA performs CAC based on the traffic parameters and the connection QoS. The CAC decides to accept or reject a new connection. The decision is based on the following questions: does the new connection affect the QoS of the connections currently being carried by the network? can the network provide the QoS requested by the new connection?

A variety of different CAC schemes have been proposed. They are classified into the following groups: equivalent capacity; heavy traffic approximation; upper bounds of the cell loss probability; fast buffer/bandwidth allocation; and time windows [13]. The CAC scheme based on equivalent capacity has better performance compared with other schemes. However, the equivalent capacity scheme makes many approximations, which result in an overestimate of equivalent capacity.

In order to make a more accurate decision for connection acceptance, we propose a fuzzy based CAC scheme, which we call Fuzzy Admission Control (FAC) scheme. The Fuzzy Logic Controller (FLC) is the main part of the FAC and its basic elements are shown in **Fig. 3**. They are the fuzzifier, inference engine, Fuzzy Rule Base (FRB) and defuzzifier. We use triangular and trapezoidal membership functions because they are suitable for real-time operation [15]. They are shown in **Fig. 4** and are given as:

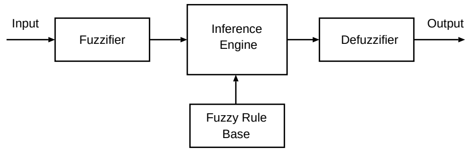

Fig. 3 FLC structure.

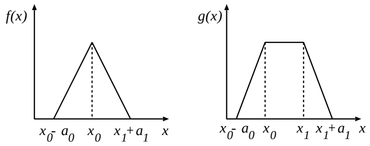

Fig. 4 Triangular and trapezoidal membership functions.

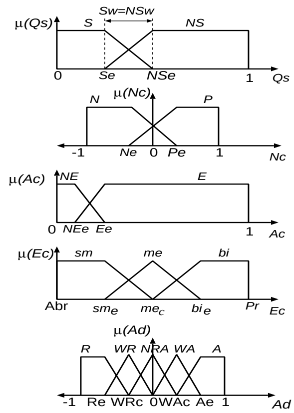

Fig. 5 FAC membership functions.

$$f(x; x_0, a_0, a_1) = \begin{cases} \frac{x-x_0}{a_0} + 1 & \text{for } x_0 - a_0 < x \leq x_0 \\ \frac{x_0-x}{a_1} + 1 & \text{for } x_0 < x \leq x_0 + a_1 \\ 0 & \text{otherwise} \end{cases}$$

$$g(x; x_0, x_1, a_0, a_1) = \begin{cases} \frac{x-x_0}{a_0} + 1 & \text{for } x_0 - a_0 < x \leq x_0 \\ 1 & \text{for } x_0 < x \leq x_1 \\ \frac{x_1-x}{a_1} + 1 & \text{for } x_1 < x \leq x_1 + a_1 \\ 0 & \text{otherwise} \end{cases}$$

where x_0 in $f(.)$ is the center of triangular function; $x_0(x_1)$ in $g(.)$ is the left (right) edge of trapezoidal function; and $a_0(a_1)$ is the left (right) width of the triangular or trapezoidal function.

The input linguistic parameters of FAC are Quality of service (Qs), Network congestion parameter (Nc), Available capacity (Ac), and user requirement parameter which is expressed by Equivalent capacity (Ec). The output linguistic parameter is the Acceptance decision (Ad). The membership functions for input and output linguistic parameters are shown in **Fig. 5**.

The small letters e and w in the membership functions means edge and width, respectively.

The term sets of Qs, Nc, Ac, and Ec are defined respectively as:

$T(Qs) = \{\text{Satisfied}, \text{NotSatisfied}\} = \{S, NS\}$;
$T(Nc) = \{\text{Negative}, \text{Positive}\} = \{N, P\}$;
$T(Ac) = \{\text{NotEnough}, \text{Enough}\} = \{NE, E\}$;
$T(Ec) = \{\text{small}, \text{medium}, \text{big}\} = \{sm, me, bi\}$.

The membership functions for input parameters of FAC are defined as follows.

$$\mu_S(Qs) = g(\log(Qs); 0, S_e, 0, S_w);$$
$$\mu_{NS}(Qs) = g(\log(Qs); Ns_e, 1, Ns_w, 0);$$
$$\mu_N(Nc) = g(Nc; -1, N_e, 0, N_w);$$
$$\mu_P(Nc) = g(Nc; P_e, 1, P_w, 0);$$
$$\mu_{NE}(Ac) = g(\log(Ac); 0, NE_e, 0, NE_w);$$
$$\mu_E(Ac) = g(\log(Ac); E_e, 1, E_w, 0);$$
$$\mu_{sm}(Ec) = g(\log(Ec); Abr, sm_e, 0, sm_w);$$
$$\mu_{me}(Ec) = f(\log(Ec); me_c, me_{w0}, me_{w1});$$
$$\mu_{bi}(Ec) = g(\log(Ec); bi_e, Pr, bi_w, 0).$$

The term set of the output linguistic parameter $T(Ad)$ is defined as {Reject, Weak Reject, Not Reject Not Accept, Weak Accept, Accept}. We write for short as $\{R, WR, NRA, WA, A\}$. The membership functions for the output parameter Ad are defined as follows.

$$\mu_R(Ad) = g(Ad; -1, R_e, 0, R_w);$$
$$\mu_{WR}(Ad) = f(Ad; WRc, WR_{w0}, WR_{w1});$$
$$\mu_{NRA}(Ad) = f(Ad; NRAc, NRA_{w0}, NRA_{w1});$$
$$\mu_{WA}(Ad) = f(Ad; WAc, WA_{w0}, WA_{w1});$$
$$\mu_A(Ad) = g(Ad; A_e, 1, A_w, 0).$$

The FRB forms a fuzzy set of dimensions $|T(Qs)| \times |T(Nc)| \times |T(Ac)| \times |T(Ec)|$, where $|T(x)|$ is the number of terms on $T(x)$. The FRB1 shown in **Table 1** has 24 rules. The control rules have the following form: IF "conditions" THEN "control action". Statements on conditions go like "the Qs is satisfied" or "the Nc is congested". Likewise, statements on control action might be "reject" or "accept".

The FAC scheme is shown in **Fig. 6**. The information for FAC are given by Bandwidth Management Predictor (BMP); Congestion Information Indicator (CII); Quality of Service Indicator (QSI); and Equivalent Capacity Estimator (ECE). The BMP works in this way: if a connection is accepted, the connection bandwidth is subtracted from the available capacity of the network, otherwise, if a connection is released, the connection bandwidth is added to the available capacity of the network. The CII decides whether the network is or isn't congested. The QSI determines whether allowing

Table 1 FRB1.

Rule	*Qs*	*Nc*	*Ac*	*Ec*	*Ad*
0	*S*	*N*	*NE*	*sm*	*NRA*
1	*S*	*N*	*NE*	*me*	*WR*
2	*S*	*N*	*NE*	*bi*	*R*
3	*S*	*N*	*E*	*sm*	*WA*
4	*S*	*N*	*E*	*me*	*NRA*
5	*S*	*N*	*E*	*bi*	*WR*
6	*S*	*P*	*NE*	*sm*	*WA*
7	*S*	*P*	*NE*	*me*	*NRA*
8	*S*	*P*	*NE*	*bi*	*R*
9	*S*	*P*	*E*	*sm*	*A*
10	*S*	*P*	*E*	*me*	*A*
11	*S*	*P*	*E*	*bi*	*A*
12	*NS*	*N*	*NE*	*sm*	*R*
13	*NS*	*N*	*NE*	*me*	*R*
14	*NS*	*N*	*NE*	*bi*	*R*
15	*NS*	*N*	*E*	*sm*	*WA*
16	*NS*	*N*	*E*	*me*	*NRA*
17	*NS*	*N*	*E*	*bi*	*R*
18	*NS*	*P*	*NE*	*sm*	*WR*
19	*NS*	*P*	*NE*	*me*	*R*
20	*NS*	*P*	*NE*	*bi*	*R*
21	*NS*	*P*	*E*	*sm*	*WA*
22	*NS*	*P*	*E*	*me*	*NRA*
23	*NS*	*P*	*E*	*bi*	*WR*

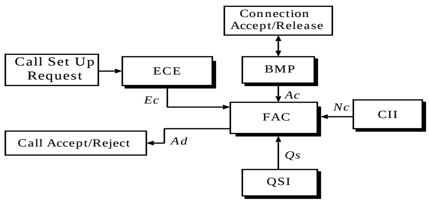

Fig. 6 FAC scheme.

a new connection violates or not the QoS guarantee of the existing connections. In this work, we consider only simple indicators for congestion and QoS. However, in order to support multimedia application, in the future, we will build more complex congestion and QoS estimators.

The ECE estimates the connection Ec. In Ref. 14), in order to get the Ec of N identical On-Off traffic sources parameter β was approximated by one. But, the assumption of $\beta \approx 1$ ignores the effect of statistical multiplexing. In order to gain from statistical multiplexing of bursty connections, we propose a Fuzzy ECE (FECE).

The FECE predicts the (Ec) required for a new connection based on the traffic parameters Peak rate (Pr), Source utilization (Su), and Peak bit-rate duration (Pbd). The term sets of Pr, Su, and Pbd are defined respectively as:

$T(Pr) = \{\text{Small}, \text{Medium}, \text{Large}\} = \{S, M, L\}$;
$T(Su) = \{\text{Low}, \text{High}\} = \{Lo, Hi\}$;
$T(Pbd) = \{\text{Short}, \text{Medium}, \text{Long}\} = \{Sh, Me, Lg\}$.

The membership functions for input parameters of FECE are defined as follows.

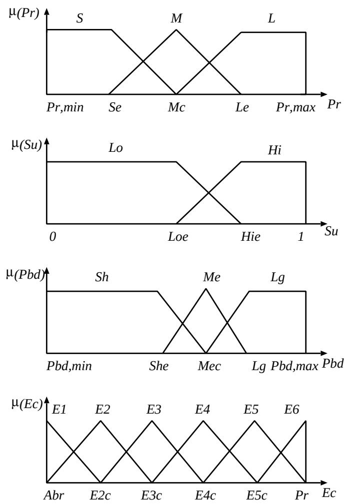

Fig. 7 FECE Membership functions.

$$\begin{aligned}
\mu_S(Pr) &= g(\log(Pr); Pr, min, S_e, 0, S_w);\\
\mu_M(Pr) &= f(\log(Pr); M_c, M_{w0}, M_{w1});\\
\mu_L(Pr) &= g(\log(Pr); L_e, Pr, max, L_w, 0);\\
\mu_{Lo}(Su) &= g(Su; 0, Lo_e, 0, Lo_w);\\
\mu_{Hi}(Su) &= g(Su; Hi_e, 1, Hi_w, 0);\\
\mu_{Sh}(Pbd) &= g(\log(Pbd); Pbd, min, Sh_e, 0, Sh_w);\\
\mu_{Me}(Pbd) &= f(\log(Pbd); Me_c, Me_{w0}, Me_{w1});\\
\mu_{Lg}(Pbd) &= g(\log(Pbd); Lg_e, Pbd, max, Lg_w, 0).
\end{aligned}$$

The Ec for a connection should fall between its Pr and Average bit rate (Abr). Therefore, we divide the Ec range in six membership functions. The term of Ec is defined as $T(Ec) = \{E1, E2, E3, E4, E5, E6\}$. The membership functions for the output parameter Ec are defined as follows.

$$\begin{aligned}
\mu_{E1}(Ec) &= f(\log(Ec); E1c, 0, E1_{w1});\\
\mu_{E2}(Ec) &= f(\log(Ec); E2c, E2_{w0}, E2_{w1});\\
\mu_{E3}(Ec) &= f(\log(Ec); E3c, E3_{w0}, E3_{w1});\\
\mu_{E4}(Ec) &= f(\log(Ec); E4c, E4_{w0}, E4_{w1});\\
\mu_{E5}(Ec) &= f(\log(Ec); E5c, E5_{w0}, E5_{w1});\\
\mu_{E6}(Ec) &= f(\log(Ec); E6c, E6_{w0}, 0).
\end{aligned}$$

The membership functions for FECE are shown in **Fig. 7** and the FRB is shown in **Table 2**.

In order to accommodate a wide variety of different traffic sources, we use the logarithmic function for some membership functions.

5. PA and DDA

5.1 SSRA

The flowchart of SSRA is shown in **Fig. 8**. The key element of SSRA is Effective Topology (ET) extraction. The ET extraction of a network is defined as the topology based on which a path is constructed for a connection.

Table 2 FRB2.

Rule	*Pr*	*Su*	*Pbd*	*Ec*
0	*S*	*Lo*	*Sh*	*E*1
1	*S*	*Lo*	*Me*	*E*2
2	*S*	*Lo*	*Lg*	*E*5
3	*S*	*Hi*	*Sh*	*E*1
4	*S*	*Hi*	*Me*	*E*1
5	*S*	*Hi*	*Lg*	*E*4
6	*M*	*Lo*	*Sh*	*E*1
7	*M*	*Lo*	*Me*	*E*3
8	*M*	*Lo*	*Lg*	*E*6
9	*M*	*Hi*	*Sh*	*E*1
10	*M*	*Hi*	*Me*	*E*2
11	*M*	*Hi*	*Lg*	*E*5
12	*L*	*Lo*	*Sh*	*E*4
13	*L*	*Lo*	*Me*	*E*6
14	*L*	*Lo*	*Lg*	*E*6
15	*L*	*Hi*	*Sh*	*E*3
16	*L*	*Hi*	*Me*	*E*5
17	*L*	*Hi*	*Lg*	*E*6

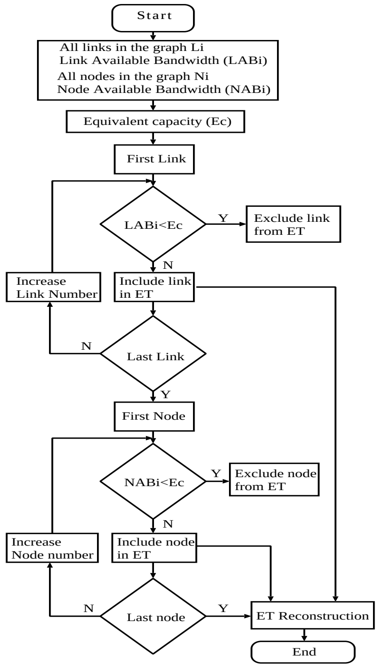

Fig. 8 SSRA flowchart.

In order to extract the ET, the network connectivity information, link and node metrics, and QoS requirement of the new connection are required. We use the Ec predicted by FECE to specify the QoS demand of a new connection. In order to have a low overhead processing time, we consider the Available Bandwidth (AB) as the only link and node metrics. If a Link Available Bandwidth (LAB) or Node Available Bandwidth (NAB) is less than Ec of a connection, this means that every path which passes via this link or node cannot satisfy the connection requirements.

First, the SSRA based on the required Ec

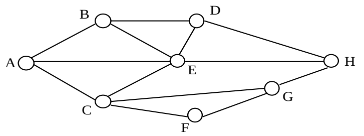

Fig. 9 Network example.

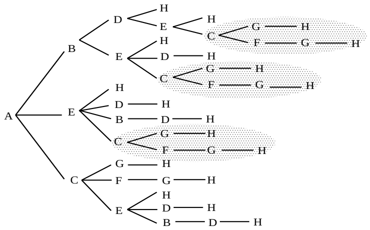

Fig. 10 Tree model.

checks all links in the network whether their AB satisfies or not the *Ec*. If a LAB doesn't satisfy the *Ec* then the link is excluded from ET. Otherwise, the link is included in the ET and the next link is checked. The procedure is repeated until all links are finished. Next, the SSRA checks all nodes in the network, whether their AB satisfies the *Ec* or not. If the NAB doesn't satisfy the *Ec* then the node is excluded from the ET. Otherwise, the node is included in the ET and the next node is checked. The procedure is repeated until all nodes are finished. Finally, after all links and nodes are checked, the network ET is constructed and the complete procedure is finished.

By using the SSRA, a network with many nodes and links will be reduced in a network with a small number of nodes and links. Thus, the proposed strategy is able to cope with more large-scale networks.

5.2 TMNA

After the execution of SSRA, the ET of the network is transformed in a tree model by TMNA. To explain this procedure, a small network with 8 nodes as shown in **Fig. 9** is considered. Node A is the Source Node (SN) and node H is the Destination Node (DN). All paths are expressed by the tree model shown in **Fig. 10**. In the shaded areas are shown the same paths from node C to H. Therefore, we further reduce the tree network as shown in **Fig. 11**. The tree model constructed by TMNA is used by IntraD agent for intra-domain routing. In the reduced tree model, each tree junction is considered as a gene and the path is represented by the chromosome.

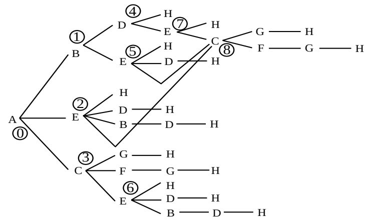

Fig. 11 Reduced tree model.

5.3 DDA

After a new connection is accepted, the RMA sends a request to the DDA. The DDA consults a table with node name entries to check whether SN and DN are in the same domain or not. If SN and DN are in the same domain, the DDA of the source domain activates the IntraD agent. Otherwise, if the SN and DN are in different domains, the InterD agent is activated.

6. RA

6.1 IntraD Agent

The IntraD agent is based on GA. A brief introduction of GA is given in following.

6.1.1 GA

GAs are search methods used to solve optimization problems. The GA mechanism is based on the interaction between individuals and the natural environment. GA comprises a set of individuals (population) and a set of biologically inspired operators (genetic operators). The individuals have genes which are the potential solutions for a problem. The genetic operators are crossover and mutation. GA generates a sequence of populations by using genetic operators among individuals. Only the most suited individuals in a population can survive and generate offsprings, thus transmitting their biological heredity to new generations [16].

GA operates through a simple cycle of four stages as shown in **Fig. 12**. Each cycle produces a new generation of possible solutions. At the first stage, an initial population is created as a starting point for the search. In the next stage, the fitness of each individual is evaluated with respect to the constraints imposed by the problem. Based on each individual's fitness, a selection mechanism chooses "parents" for the crossover and mutation operators. The crossover operator takes two chromosomes and swaps part of their genetic information to produce new chromosomes. The mutation operator introduces new genetic structures in the population by randomly modifying some of genes,

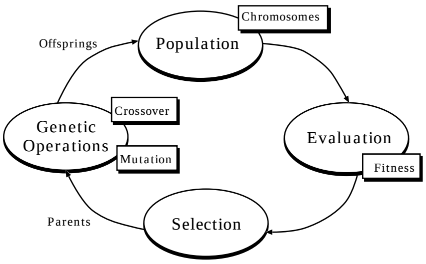

Fig. 12 GA cycle.

helping the search algorithm to escape from local minima's traps. The offsprings produced by the genetic manipulation process are the next populations to be evaluated. GA can replace either a whole population or its less fitted members only. The creation-evaluation-selection-manipulation cycle repeats until a satisfactory solution is found or the termination criterion is achieved.

6.1.2 IntraD Algorithm

The IntraD algorithm is a delay-constraint unicast source routing mechanism and is based on GA. The most important factor to achieve efficient genetic operations is gene coding. In the Genetic Load Balancing Routing (GLBR) algorithm [7], the genes are put in a chromosome in the same order the nodes are in a path, so the chromosomes have different sizes which result in complex crossover operation. Also, the GLBR algorithm may enter in routing loops. Furthermore, when the genetic operations are chosen randomly, the new offsprings of a population (paths) may not exist, as a result, the GLBR algorithm should check the validity of the searched path. If the searched path doesn't exist, other genetic operations should be carried out in order to find a new path.

In order to simplify the genetic operations of GLBR, in the IntraD algorithm, the network is expressed by a tree network and the genes represent the tree junctions. A chromosome example is shown in **Fig. 13**. The genes in a chromosome have two states "active" and "inactive". A gene is called "active" if the junction is in the path, otherwise the gene is in "inactive" state. The genetic operations are carried out in the "active" genes. Each gene includes information of the adjacent nodes. The paths are represented by chromosomes which have the same length. Therefore, the crossover operation becomes very easy. In GLBR algorithm, the interaction between the adjacent genes in a chromosome is necessary. On the other hand,

0	1	2	3	4	5	6	7	8
BEC	DE	HD BC	GFE	HE	HDC	HDB	HC	GF

Fig. 13 A chromosome example.

in the IntraD algorithm this interaction is not necessary. So, the mutation operation becomes easy.

The IntraD algorithm selection operation uses both the ranking and elitist models. The ranking model ranks each individual by their fitness. The rank is decided based on the fitness and the probability is decided based on the rank. The individual fitness is based on the path delay time. If the delay time is small, the individual fitness is high. When the rank is high, the probability of individuals is high. In the elitist model, the individual which has the highest fitness value in a population is left intact in the next generation. Therefore, the best value is always kept and the routing algorithm can converge very fast to the desired delay time.

As the crossover method is used the single point crossover. The crossover point is selected in the same locus of two selected individuals. In the mutation operation the genes are chosen randomly in the range from zero up to mutation probability $p\_mutation \leq \frac{1}{\ell}$, where l is the chromosome length.

By using the tree model, the IntraD algorithm is able to avoid routing loops. Also, the searched path always exists, so the algorithm doesn't need to check the validity of the searched path.

6.2 InterD Agent

After the DDA finds out that SN and DN are in different domains, the InterD agent is activated. The InterD agent is a composition of IntraD agent and CMA. It use an escalation strategy to make the inter-domain routing. By using the escalation strategy, the information exchange is needed only in domains where the selected path passes. Thus, the information flooding in all domains is not necessary and the network resources can be use efficiently. The InterD agent operates in the following way. After receiving a connection request, a node become a SN. The IntraD agent finds a path inside the domain. The DN of the source domain starts the CMA. The CMA is a simple agent. It finds the best link by using a sorting algorithm based on the inter-domain links parameters. After the CMA decides the best link for connection, the DN of this link becomes a SN and the IntraD agent is activated in the following domain. This procedure is escalated until the DN of the destination domain is found.

7. Simulation Results

In this section, we evaluate the performance behavior of FECE and IntraD algorithm. Additional work is in progress to evaluate the total performance of the proposed strategy.

7.1 Ec Estimation

Considering a two-state Markov source the expressions of Ec for exact value, flow approximation and stationary approximation are given in Ref. 14). Assuming a finite Buffer (B) size, the equation satisfied by the Ec for an overflow probability of ϵ is given by:

$$\epsilon = \beta \cdot \exp\left(-\frac{B(Ec - Su \cdot Pr)}{Pbd(1 - Su)(Pr - Ec)Ec}\right) \tag{1}$$

where,

$$\beta = \frac{(Ec - Su \cdot Pr) + \epsilon \cdot Su(Pr - Ec)}{(1 - Su)Ec}. \tag{2}$$

If the parameter β is approximated by 1, the Ec for a single connection is given by:

$$\hat{E}c \approx \frac{\alpha \cdot Pbd(1 - Su)Pr - B}{2\alpha \cdot Pbd(1 - Su)} + \frac{\sqrt{[\alpha \cdot Pbd(1 - Su)Pr - B]^2 + 4B\alpha \cdot Pbd \cdot (1 - Su)Pr}}{2\alpha \cdot Pbd(1 - Su)} \tag{3}$$

where $\alpha = \ln(1/\epsilon)$.

For multiple connections, when the input bit rate is characterized by a N-state Markov chain, the distribution of the buffer contents is of the following form:

$$F(B) = \sum_{i=1}^{N} a_i \Phi_i e^{Z_i B} \tag{4}$$

where Z_i and Φ_i are, respectively, generalized eigenvalues and eigenvectors associated with the solution of the differential equation satisfied by the stationary probabilities of the system, and a_i are coefficients determined from boundary conditions.

The exact value of the Ec for single and multiple connections are calculated by iteratively solving Eqs. (2) and (4). But, this calculation, although exact, is complicated and is not compatible with a dynamic and real-time environment [14].

The Ec for multiple connections using flow approximation is calculated by:

$$\hat{E}c_{(F)} = \sum_{i=1}^{N} \hat{E}c_i \tag{5}$$

where $\hat{E}c_i$ are determined from Eq. (3).

In the flow approximation the parameter β is considered 1. This approximation can do a good evaluation in the case when either Number (Nr) of connections is small of the actual total Ec is close to overall Abr. In other cases, this approximation results in an overestimate of Ec.

When Nr connections with relatively long burst periods are multiplexed, a reasonably accurate estimate of the required Ec can be obtained from the stationary approximation. The value of the Ec can be expressed as:

$$\hat{E}c_{(S)} \approx Abr + \hat{\alpha}\sigma \tag{6}$$

where Abr is the average aggregate bit rate ($Abr = \sum_{i=1}^{N} Abr_i$); $\hat{\alpha}$ is $\sqrt{-2\ln(\epsilon) - \ln(2\pi)}$, and σ is the standard deviation of the aggregate bit rate ($\sigma^2 = \sum_{i=1}^{N} \sigma_i^2$).

The stationary approximation gives a substantial overestimate of the Ec because it ignores the effect of the buffer.

The parameter values of input membership functions for FECE are assigned as follows. For Pr, $S_e = -3$, $S_w = 1$, $M_c = -2$, $M_{w0} = M_{w1} = 1$, $L_e = -1$, $L_w = 1$, $Pr, min = 10^{-4}$, $Pr, max = 1$; for Su, $Lo_e = 0.6$, $Lo_w = 0.15$, $Hi_e = 0.75$, $Hi_w = 0.15$; for Pbd, $Sh_e = -3$, $Sh_w = 1$, $Me_c = -2$, $Me_{w0} = Me_{w1} = 1$, $Lg_e = -1$, $Lg_w = 1$, $Pbd, min = 10^{-9}$, $Pbd, max = 100$s.

The parameter values of output membership functions for FECE are assigned as follows. The value of Ec_1 is set equal to Abr and the value of Ec_6 is set equal to Pr. The other values are calculated based on the following equation:

$$C_i c = C_{i-1} c + (Pr - Abr)/5 \tag{7}$$

where i = 2, 3, 4, 5, 6.

Considering the same parameters for the four methods: $Pr = 10$ Mb/s, $Pbd = 0.02$ s, the probability of overflow 10^{-5}, the characteristic of the required equivalent capacity versus source utilization for the number of connections $Nr = 50$ is shown in **Fig. 14**. The required Ec calculated by FECE is very close to the exact value. For bursty traffic sources when the sources have a low utilization, the flow approximation hasn't a good Ec accuracy. But, for traffic sources with high source utilization, the flow approximation has a good Ec estimation. On the other hand, the stationary approximation has a good Ec accuracy for low source uti-

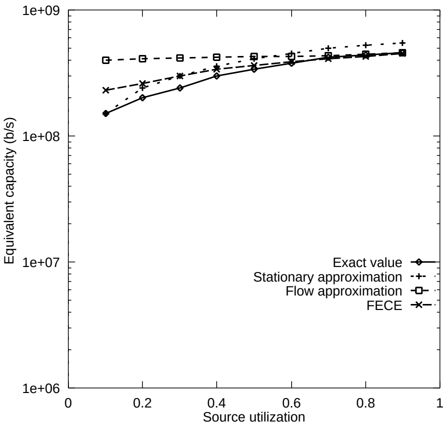

Fig. 14 Equivalent capacity versus source utilization.

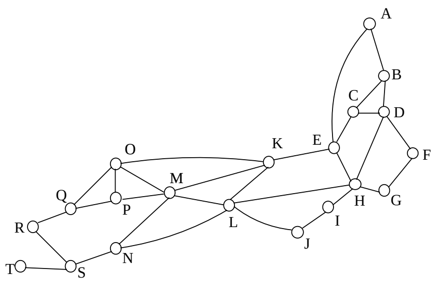

Fig. 15 Network model.

lization and a poor estimation for high source utilization. The characteristic of FECE is more close to exact value compared with both flow and stationary estimations. But for sources with very low utilization, the stationary approximation give good accuracy than FECE. Therefore, it is possible to get a better estimation of Ec if we calculate the value of Ec as the minimum value of FECE and stationary approximation.

7.2 Routing Algorithms Performance

In order to make the comparison with Ref. 7), we consider that, the network used for simulation in **Fig. 15** is the ET extracted by SSRA. In Ref. 7) is shown a comparison performance between the GLBR, SPF and RIP algorithms. The GLBR algorithm has a better behavior compared with SPF and RIP algorithms. Therefore, in following, we compare the performance of IntraD algorithm with GLBR algorithm.

To compare both algorithms, the first population is selected the same. After the congestion situation happens in path currently in use, the IntraD and GLBR algorithms search for a new path in order to avoid the congested path. The genetic operations are repeated until the path with smallest delay is found or the initialized generation size is achieved. The parameters used in simulations are shown in **Table 3**. Based on the data obtained from the IntraD and GLBR algorithms, the characteristics of delay time versus simulation step are depicted.

Table 3 Simulation parameters.

population size	5, 10, 20, 30
crossover rate (%)	70, 80, 90, 100
mutation rate (%)	1, 5, 10, 20

The IntraD and GLBR algorithms are compared for different population sizes, crossover rates and mutation rates, but for the sake of space, we have shown in **Table 4** only the simulation data for population sizes 10 and 20. The mutation rate changes from 1% to 20% and the crossover rate changes from 70% to 100%. The values inside the table show the search rate when both algorithms find the shortest path. In all simulations, the IntraD algorithm finds the shortest path faster than the GLBR algorithm.

For the population size 5, the search rate was high. This means that number of genetic operations to find the shortest path increases. For the population size 10, the result was improved, and when the population size was 20 the result was improved much more. However, when the population size was 30, both algorithms could not achieve an efficient search, because the genetic operations become very complex. We conclude that, the best population size is 20. The decision of the best population size is a tradeoff between diverse constrains. If the population size is small, the algorithms converge fast to a local minima, but the algorithms may give not the best response. Otherwise, if the population size is big, the algorithms need time to carry out the genetic operations. The change of crossover rate doesn't have too much effect on the results of algorithms. On the other hand, the change of mutation rate has a great effect in the algorithms performance. If the mutation rate is small, the created population types are limited. Otherwise, if the mutation rate is big, the delay time doesn't decrease. Therefore, the algorithms need time to find the shortest path. We conclude that a mutation rate of about 10% is a good mutation rate.

Figure 16 shows the characteristics of delay time versus simulation step for the IntraD and GLBR algorithms. The simulation step con-

Table 4 Search rate (%) of the IntraD and GLBR algorithms.

population size 10								
mutation	crossover rate 70%		crossover rate 80%		crossover rate 90%		crossover rate 100%	
rate	IntraD	GLBR	IntraD	GLBR	IntraD	GLBR	IntraD	GLBR
1%	49.5	52.0	46.2	54.2	49.5	56.4	44.6	47.6
5%	36.3	44.0	40.9	43.6	40.8	46.6	39.3	41.1
10%	27.7	37.1	31.7	41.1	30.9	33.9	31.5	39.1
20%	30.5	35.9	27.3	33.9	19.9	32.9	20.4	28.1
population size 20								
mutation	crossover rate 70%		crossover rate 80%		crossover rate 90%		crossover rate 100%	
rate	IntraD	GLBR	IntraD	GLBR	IntraD	GLBR	IntraD	GLBR
1%	17.4	21.1	14.9	21.3	18.6	21.3	17.4	20.9
5%	13.2	16.8	16.6	23.7	14.7	18.7	17.8	18.4
10%	16.9	18.7	16.1	20.9	17.9	26.7	15.6	25.7
20%	12.9	15.9	13.4	21.2	12.6	22.3	15.9	26.7

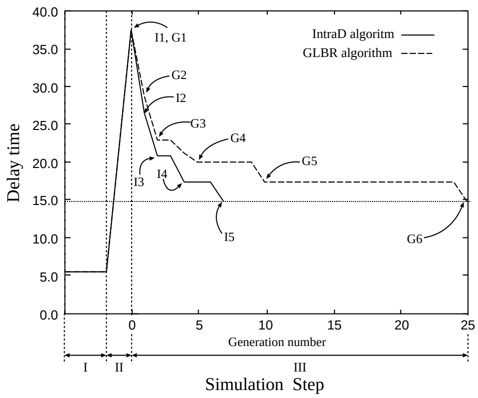

Fig. 16 Performance comparison of IntraD and GLBR algorithms.

sists of three parts: step I is the communication state, step II is the congestion state and step III is the algorithm operation state. Step III (generation size) shows how many genetic operations are needed in order to find the shortest path. The IntraD algorithm can find the shortest path faster than the GLBR algorithm. The GLBR genetic operations complexity is because new individuals (paths) may not exist, so the GLBR algorithm should generate new populations to get the shortest path.

Table 5 shows a comparison between two algorithms for different Generation Number (GN). The labels inside the table are obtained in the points where the delay time has changed. The labels I2–I5 are for IntraD algorithm and labels G2–G6 are for GLBR algorithm. In labels I1 and G1 the GN is zero. These points are the points where both algorithms start the search. The search results are shown up to the rank number 7. The IntraD algorithm has achieved the rank number 7 after 7 generations. The selected route is "ABDCEHLNST". While in this stage, the GLBR algorithm is in the route "ABDHLNMPQRST" and the rank number is 36. The GLBR algorithm needs 24 generations for the rank number 7.

The InterD agent uses an escalation strategy for routing. The total time needed to find a feasible path is an additive function and depends on the number of domains and the speed of interdomain links. Considering very high-speed interdomain links, the total time to find a path will be approximately equal with the sum of IntraD algorithms execution time.

8. Future Work

The authors are planing to extend the proposed strategy in the following directions.

Parallel GA Implementation of IntraD Agent—We intend to implement the IntraD agent in a parallel GA architectures [17),18)]. By implementing the IntraD agent in a parallel GA, the proposed strategy can be a good candidate for high-speed large-scale networks.

Load Distribution—We considered the routing problem only for the case when a congestion or failure situation happens in the route currently in use. We plan to extend the study for distribution of the load in different routes.

Multimedia Application—The proposed routing algorithm is an unicast delay-constrained algorithm, which uses the delay as the only QoS constrain. For multimedia application, we plan to develop the proposed strategy to find a path which satisfies multiple QoS constrains such as bandwidth, delay and loss probability.

9. Conclusions

In this paper, we proposed an integrated CAC and routing strategy using cooperative agents. The proposed strategy have the following characteristics.

Table 5 Comparison for different GN.

GN	IntraD				GLBR			
	Label	Route	Delay	Rank	Label	Route	Delay	Rank
1	I2	ABDCEHLNMPOQRST	26.5	175	G2	ABDCEHIJLNMPOQRST	28.9	203
2	I3	ABDHLKMPOQRST	20.9	54				
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
4	I4	ABDHLKMPQRST	17.3	19	G3	ABDHEHLNMPQRST	21.2	69
5	⋮	⋮	⋮	⋮	G4	ABDHLNMPQRST	20.0	36
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
7	I5	ABDCEHLNST	14.7	7	⋮	⋮	⋮	⋮
⋮					⋮	⋮	⋮	⋮
10					G5	ABDHLKMPQRST	17.3	19
⋮					⋮	⋮	⋮	⋮
24					G6	ABDCEHLNST	14.7	7

Integration of CAC and Routing—The proposed strategy integrates CAC and routing.

Combination of Source and Distributed Routing—The proposed routing algorithm is a combination of source and distributed routing. It uses source routing inside a domain and hop-by-hop routing for inter-domain.

Reduction of Search Space—By using SSRA, the proposed strategy is able to reduce the search space and give a fast decision.

Adaptive, Flexible and Intelligent—In order to be adaptive flexible and intelligent, the proposed strategy is based on DAI approach.

Avoid Flooding—In order to avoid flooding, the InterD agent uses an escalation strategy which transmits the information only to domains in which the routing path passes.

Avoid Rooting Loops—By using TMNA, the proposed strategy is able to avoid the rooting loops.

Is Scalable—The proposed strategy can be scaled-up very easily by increasing the number of domains and agents.

We carried out some simulations to evaluate the performance of the FECE and IntraD algorithm. From the simulation result, we conclude:

- the FECE has a good Ec estimation compared with conventional methods;
- combination of FECE and stationary approximation will result in a more accurate estimation of Ec;
- the IntraD algorithm has a better performance compared with GLBR algorithm;
- the IntraD algorithm uses a novel gene coding method, therefore has an efficient search;
- the GLBR genetic operations are more complex than the IntraD algorithm operations.

References

1) Lin, F.Y.S. and Yee, J.R.: A Real Time Distributed Routing and Admission Control Algorithm for ATM Networks, *Proc. IEEE INFOCOM '93*, pp.7a.1.1–7a1.10 (1993).

2) Baransel, C., Dobosiewicz, W. and Gburzynski, P.: Routing in Multihop Packet Switching Networks: Gb/s Challenge, *IEEE Network*, Vol.9, No.3, pp.38–60 (1995).

3) Lee, W.C., Hluchyj, M.G. and Humblet, P.A.: Routing Subject to Quality of Services Constraints in Integrated Communications Networks, *IEEE Network*, Vol.9, No.4, pp.46–55 (1995).

4) Wang, Z. and Crowcroft, J.: Quality of Services Routing for Supporting Mulimedia Application, *J. Select. Areas Commun.*, Vol.14, No.7, pp.1228–1234 (1996).

5) Zhu, T., Liu, C. and Mouftah, H.T.: Dynamic Routing for Multimedia Traffic over ATM Networks, *Proc. IEEE Symposium on Computer and Commun.*, pp.91–96 (1995).

6) Tanenbaum, A.S.: Computer Networks (Third Edition), Prentice Hall PTR (1996).

7) Munetomo, M., Takai, Y. and Sato, Y.: An Adaptive Routing Algorithm with Load Balancing by a Genetic Algorithm, *Trans. Information Processing Society of Japan*, Vol.39, No.2, pp.219–227 (1998).

8) Cheng, R. and Chang, C: Design of a Fuzzy Traffic Controller for ATM Networks, *IEEE/ACM Trans. Networking*, Vol.4, No.3, pp.460–469 (1996).

9) Khalfet, J. and Chemouil, P.: Application of Fuzzy Control to Adaptive Traffic Routing in Telephone Networks, *Information and Decision*

Tech., Vol.19, No.4, pp.339–348 (1994).
10) Habib, I. (Ed.): Neurocomputing in High-Speed Networks, *IEEE Commun. Magazine*, Special Issue, Vol.33, No.10 (1995).
11) Barolli, L., Koyama, A., Yokoyama, S. and Yamada, T.: An Intelligent Policing-Routing Mechanism Based on Fuzzy Logic and Genetic Algorithms, *Proc. ICPADS '98*, pp.390–397 (1998).
12) Gaiti, D. and Boukhatem, N.: Cooperative Congestion Control Schemes in ATM Networks, *IEEE Commun. Magazine*, Vol.34, No.11, pp.102–110 (1995).
13) Perros, H.G. and Elsayed, K.M.: Call Admission Control Schemes: A Review, *IEEE Commun. Magazine*, Vol.35, No.11, pp.82–91 (1996).
14) Guärin, R., Ahmadi, H. and Naghshineh, M.: Equivalent Capacity and Its Application to Bandwidth Allocation in High-Speed Networks, *IEEE J. Select. Areas Commun.*, Vol.9, No.7, pp.968–981 (1991).
15) Dubois, D., Prade, H. and Yager, R. (Eds.): *Fuzzy Sets for Intelligent Systems*, Morgan Kaufman Publishers (1993).
16) Goldberg, D.E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Company (1989).
17) Cantu-Paz, E.: Designing Efficient and Accurate Parallel Genetic Algorithms, Ph.D. Thesis, University of Illinois at Urbana-Champaign (1999).
18) Tang, K.S., Man, K.F. and Kwong, S.: Parallel Genetic Algorithms: Implementable Hardware Solutions, *IEEE Circuits and Systems Society Newsletter*, Vol.10, No.2, pp.3–11 (1999).

(Received April 1, 2000)
(Accepted October 6, 2000)

Leonard Barolli was born in Bilisht, Albania. He received the B.E. and Ph.D. degrees in 1989 and 1997 from Tirana University and Yamagata University, respectively. From April 1997 to March 1999, he was a Post Doctor Fellow Researcher of JSPS at Faculty of Engineering, Yamagata University. From April 1999, he is working as a Research Associate at Department of Public Policy and Social Studies, Yamagata University. His research interests include network traffic control, fuzzy control, genetic algorithms and agent-based systems. He is a member of SOFT and IPSJ.

Akio Koyama was born in Yonezawa, Yamagata prefecture. He received the B.E. and Ph.D. degrees from Yamagata University in 1987 and 1998, respectively. From April 1981 to March 1999, he was working as a technical staff at Faculty of Engineering, Yamagata University. From April 1999, he is working as an Assistant Professor at Department of Computer Software, the University of Aizu. His current research interests include network agent systems, high-speed network protocols, distance learning systems, and network traffic control. He is a member of IEEE Computer Society, IPSJ and IEICE.

Takako Yamada was born in Osaka. She received the B.Ec. in 1983 from Tohoku University and M.S. and Ph.D. degrees from Tokyo Institute of Technology in 1986 and 1995, respectively. From April 1995, she is working as an Associate Professor at Department of Public Policy and Social Studies, Yamagata University. Her research interests include modeling and performance evaluation, telecommunication networks, and mobile communication. She is a member of Operational Research Association of Japan, IPSJ and IEICE.

Shoichi Yokoyama was born in Marugame, Kagawa prefecture. He received the B.E. and Ph.D. degrees from University of Tokyo in 1972 and 1987, respectively. In 1972 he was with the Electrotechnical Laboratory, Information Science Division. Since 1993 he has been a Professor at Faculty of Engineering, Yamagata University. His current research interests include natural language processing, electronic dictionary, machine translation, and artificial intelligence. He is a member of IPSJ, SICE, JASJ, ACL and Euralex.