Eternity's Team Description, Robocup Rescue Simulation

Ehsan Mahmoudy, Siavash Mirarab, Niusha Hakimipour, Ali Akhavan Bitaghsir

Department of Electrical and Computer Engineering
Tehran university, Iran, Tehran

{sehsanm, s.mirarab, n.hakimipour, a.akhavan}@ece.ut.ac.ir

Abstract. The RoboCup Rescue Simulation domain [1] , is a challenging multi agent environment , in witch tasks need to be assigned collaboratively to different types of agents . In this domain, achieving the main goal requires adaptive high performance Cooperation of several agents. The challenging problems in Rescue Simulation are: (1) Lack of information from the environment (Only local information is available for each agent), (2) The dynamic environment makes the available information unreliable after a period of time, (3) Communication between the agents is limited , so its precious to transmit as much data as possible in low size messages. In this paper, We'll show our methods to deal with these bottlenecks in the system.

1 Introduction

The RoboCup Rescue Simulation domain [1] , is a challenging multi agent environment , in witch tasks need to be assigned collaboratively to different types of agents . In this domain, achieving the main goal requires adaptive high performance Cooperation of several agents. The challenging problems in Rescue Simulation are:

- Lack of information from the environment (Only local information is available for each agent).
- The dynamic environment makes the available information unreliable after a period of time
- Communication between the agents is limited, so it's precious to transmit as much data as possible in low size messages.

These problem lead us to design an adaptive, stigmergic and cooperative system based on Ant Colony Optimization (ACO). The main idea behind our system is inspired from AntNet originally proposed by M.Dorige and G.dicaro in 1997. In AntNet routing, a group of mobile agents (or artificial ants) build paths between pairs of nodes exploring the roads network concurrently and exchanging obtained information in order to update the routing tables.

The traditional routing methods do not have enough flexibility and adaptivity in dynamic environments like Rescue simulation .

2 Ant Colony Routing Algorithm

In the Ant based routing algorithms, routing is determined by means of very complex interaction of forward and backward city roads network exploration agents (Ants). The idea behind this subdivision of agents is to allow the backward ants to utilize the useful information gathered by the forward Ants, on their trip form source to destination. Suppose we have N nodes (in the city roads network), where s denotes a generic source node, when an agent (Ant) is generated toward a destination d, from this node. Two types of ants are explained:

- Forward Ant ,denoted $F_{s \rightarrow d}$,which travels from the source node s to destination d.

- Backward Ant ,denoted $B_{s \rightarrow d}$,will be generated by a forward ant $F_{s \rightarrow d}$ in the destination .It will come back to s following the same path traversed by $F_{s \rightarrow d}$, with the purpose of using the information already picked up by $F_{s \rightarrow d}$ in order to update routing tables of the visited nodes.

Every ant transports a Stack $S_{s \rightarrow d}(k)$ of data , whether k index refers to the k_{th} visited node , in a journey, where $S_{s \rightarrow d}(0)=s$ and $S_{s \rightarrow d}(m)=d$ being m the amount of jumps performed by $F_{s \rightarrow d}$ for arriving to d. Let k be any network node; its routing table will have N entries, one for each possible destination.Let j be one entry of k routing table (a possible destination) and Let N_k be set of neighboring nodes of node k. Let P_{ji} be the probability with which an ant or data packet in k, jumps to a node i, $i \in N_k$, when the destination is j ($j <> k$). Then, for each of the N entries in the node k routing table , it will be n_k values of P_{ji} subject to the condition:

$$\sum_{i \in N_k} P_{ji} = 1 \mbox{ ; } 1 \leq \forall j \leq N \qquad \textbf{(1)}$$

The following lines show AntNet1.0[3] pseudocode,using the symbols and nomenclature already presented:

```
BEGIN
{

    Routing Tables Set-Up: For each node k
     the routing tables are initialized with
     a uniform distribution probability:
      left( P_{ji} = frac{1}{n_k} and : forall i in N_k right)
     DO always (in parallel)
     {

          STEP 1: In regular time intervals, each
          node s launches an F_{s → d} ant
          to a randomly chosen destination d.
          /*when F_{s → d} reaches k, (k <> d},it performs step 2*/
          DO (in parallel, for each F_{s → d})

               STEP 2:  F_{s → d} pushes in its
               Stack S_{s → d}(k) node k identifier
               and the time between its launching from s
                 to its arriving to k.
               F_{s → d} selects the next node in
```

```
               two ways:
(a) It draws between i nodes,
                         i in N_k, where   each node
                          i  has  a  P_{di}   probability
       (in the k routing table) to be selected.
       {bold IF} the node selected in (a) was alrea-
       dy visited.
 (b)  It draws again the  jumping  node,
          but now with the same  probability
          for all neighbors i, i in N_k.
          {bold IF} the selected node was already   visited
    STEP 3 : A cycle is found.F_{s → d}
    pops from its stack all data of the
    cycle nodes, because the optimal
    path must not have any cycle.F_{s→ d}
    returns to 2 (a) if the time spent in
    the cycle is less than a threshold;
    else it dies in order to avoid infinite
    loops.
 END IF
 END IF

     WHILE jumping node <> d
     STEP 4: F_{s → d} generates in d
     another ant, called backward ant
     B_{s → d}. F_{s→ d}  trans-
     fers to B_{s → d} its stack.
   /*B_{s → d}, will return to s, following
     the path used by F_{s → d}*/
    DO (in parallel, for each B_{s → d} ant)
   {
         /*When B_{s → d} arrives from a node
             f, f in N_k to a node k, it performs
             step 5*/
          STEP 5: B_{s→ d} updates the k rou-
          ting table and list of trips, for the
          entries regarding to nodes k' bet-
         ween k and d inclusive, according
          to the data carried in S_{s → d}(k'),
          Increasing probabilities associated
```

```
          to path  used  and decreases other
         paths probabilities , by  mean of  a
        criteria explained in [2].
        IF k <> s
               B_{s → d} will leave k and jump
                To a node given by S_{s → d}k-1).
       END IF
  WHILE (k <> s)
END
```

The routing table and list of trips updating methods for k are described as follows:

Routing table of node k is updated for the entries corresponding to all nodes k' between k and d inclusive. For example, the updating approach for the d node, when $B_{s \rightarrow d}$ arrives to k, coming from f, f in N_k is briefly explained, as following:

- A P_{df} probability associated with the node f when it wants to update the data corresponding to the d node is increased, according to:

$$P_{df} \leftarrow P_{df}+(1-r')*(1-P_{df}) \qquad (2)$$

where r' is an a dimensional measure, indicating how good (small) is the elapsed trip time T with regard to what has been observed on average until that instant. Experimentally, r' is expressed as:

$$r' = \frac{T}{c\mu} \mbox{ (c geq 1 if } \frac{T}{c\mu} < 1\mbox{) Or 1 o.w} \qquad (3)$$

where: mu is average of the observed trip-time T ;
and. c is a scale factor experimentally chosen as 2 [2]. More details about r' and its significance can be found in [2].

- The other neighboring nodes $(j <> f)$ P_{dj} probabilities associated with node k are diminished, in order to satisfy equation (1),through the expression:

$$P_{dj} \leftarrow P_{dj}-(1-r')*P_{dj} \quad \mbox{ forall } j \in N_k, j <> f \qquad (4)$$

A list $trip_k(\mu_i,\sigma_{i}^{2})$ of estimate arithmetic mean values μ_i and associated variances σ_{i}^{2} for trip times from node k to all nodes i i <> k is also updated. This data structure represents a *memory* of the network state as *seen* by node k . The list trip is updated with information carried by $B_{s \rightarrow d}$ ants in their stack $S_{s \rightarrow d}$. For any node pair source-destination, mu after (n+1) samples (n>0) is calculated as follows:

$$\mu_{n+1} = \frac{n\mu_n + x_{n+1}}{n+1} \qquad (5)$$

where: x_{n+1} trip time T sample n+1, and mu_n is the arithmetic mean after n trip time samples.

Before going to the details of the system is the domain of Rescue Simulation , we explain a basic idea behind the system which is Partitioning.

3 Partitioning

From the algorithm mentioned above, we imply that for Ant based routing, time and memory consuming processes over all nodes of the city is required in our agents. In the 1/1 scale of Rescue Simulation the number of roads is approximately 10000 which is hard to handle. so we decided to divide the city into several parts and then each agent is responsible for it's specific part of city. Benefits of this idea are:

- Less time and memory consumption, because processes are done simultaneously on several agents on smaller parts of the city.
- Easier agent management and cooperation.
- Each agent has more reliable data at each time. because it's data is more uptodate in it's small and specific part .

We use a simple partition which has "strong connectivity" for each part of the environment.
Going to the detail of our system we describe the system in two stages discussed in the following section:

4 The Two Stages

4.1 Stage I

This stage is devoted to the first cycles of the simulation. At the beginning of simulation each agent's data is local; therefore the agent has a raw data about the global environment. After some cycles, each agent gathers more information about it's region. This is why we separated our strategy into two stages.

We Propose some roads as "good" roads. These strategic roads are used by different agents during the simulation as representative compact characteristics of each region used in agents for routing.

For Finding these "good" roads we use an Ant Colony based algorithm in each agent. For this purpose we compute the average of probabilities of entrance into each road *in the routing tables updating by the routing system*; then "good" roads are defined as roads with highest average.

Police Forces consider high priority for clearing these roads. As the raw and basic information of agents about the unknown regions are nearly the same, "good" roads chosen by different agents are similar. Therefore if police forces clear these roads then the other agents can use them easily.

The other usage of "good" roads appear when an agent wants to go across a region and hasn't enough updated information about that region to use an ant based routing.

Each agent use it's specific region as the domain of Ant based routing with some modification s which are:

- At regular intervals, each agent produces virtual ants lunched from several nodes of its region r, also the ants destinations are selected inside r.
- In the original AntNet algorithm ,the trip time of each virtual ant, was the effective factor for updating the routing tables. In our system each virtual ant A computes it's own trip time as a function defined below :

$$T_{trip}(A) = \sum_{R \in A\text{'s path}} T_{road}(R) \mbox{ ;in which A is an Ant} \qquad (6)$$

$$T_{road}(R) = \frac{\alpha}{PLH(R)} + \frac{\beta}{d(R)} \qquad (7)$$

Where PLH(R) is the number of lines to the head of road R, which are passable; obtained from the formulas listed below (extracted from RCRSS Manual Ver 1.0 [1]) :

$$lineWidth = \frac{width}{linesToHead+linesToTail} \qquad (8)$$

$$blockedlines = \lfloor \frac{block}{2.lineWidth}+0.5 \rfloor \qquad (9)$$

$$passableLinesToHead = \max(0, linesToHead - blockedLines) \qquad (10)$$

$$passableLinesToTail = \max(0, linesToTail - blockedLines) \qquad (11)$$

in which d(R) is the distance of road R from the nearest fire site.
also alpha and beta are coefficients showing the importance of each factor (PLH(R) and d(R)).

In this stage, in each region the police forces try to clear some good roads which are likely to form a spanning tree in the region. The main benefit behind this idea is:

The roads selected for clearing in this stage are more probable to be used by other agents in the next cycles of simulation, when they want to go across the region .

We found out that ambulance team agents don't need to change their region frequently, so we selected this type of agent as the storage place, where the information of one region is stored for the usage of requesting agents.

Every time an agent wants to pass across a region (which is not it's own region) the ambulance in that region transmits the good roads of the region to that agent; in case the connection between them can't be be established (because of far distance or high communication load between agents) the agent applies the Ant based routing with it's raw information from the region .

4.2 Stage II

After clearing good roads by police forces, the second stage begins . In this stage each agent, at each time, may have a destination node d . if d is in it's own region , the agent uses ant based routing by it's up-to-date information of the region. Otherwise suppose d is in region r_d, the agent asks it's center for sequence of regions, starting from it's own region to region r_d .The center agent outputs the requested path by means of a simple routing Considering the region's and the roads connecting them in the map . After getting the path, the agent crosses the regions in the path one by one; for this purpose, the agent requests the ambulance which is responsible for the region in order to send it a suitable route through the region and also a list of good roads described before. The agent uses these good roads when the route is not passable at the time agent wants to pass (unreliable data). it is notable that "good" roads are not fixed (because the environment is dynamic) but updating by the ambulances frequently. These updated "good" roads are then transmitted to police forces without any specific destination(wandering polices without any critical task) , so that they can clear these new roads for future usage .

5 Conclusion

In this Team Description, We've discussed about the methods used in the Robocup Rescue Simulation [1] contest, as a basis for constructing the team – Eternity. We're testing whether these methods will lead us into a good situation or not. The big problem about the subject, is the hardship of its implementation as a Robocup Rescue Simulation Team, which will work under this domain, confronting to all of the standards defined in it.

References

1. Takeshi Morimoto - edited by RoboCupRescue Technical Committee : How to Develop a RoboCupRescue Agent for RoboCupRescue Simulation System Version 0.1st edition, 2002.
2. M.Dorigo, G. Di Caro : AntNet.Distributed Stigmergetic for Communication Networks. Journal of Artificial Intelligence Research. Vol. 9 , 1998
3. Gianni Di Caro, Marco Dorigo : Two Ant Colony Algorithms for Best-Effort Routing in Datagram Networks
4. I.Kassabalidis, M.A, E1-Sharkawi, R.J.Marks II, P.Arabshahi, A.A.Gray : Swarm Intelligence for Routing in Communication Networks
5. Y.Yang, A.N.Zinicir-Heywood, M.I.Heywood, S.Srinivas : Agent-Based Routing Algorithms on a LAN.
6. Benjamin Baran , Ruben Sosa : AntNet Routing Algorithm for Data Networks based on Mobile Agents , 2000.