

Tree Building Control Protocol – State Machine



Ondřej Dolejš
*Department of Control Systems
Center for Applied Cybernetics
Faculty of Electrical Engineering
Czech Technical University
Karlovo nám. 13, 121 35 Prague 2
Czech Republic
Tel : +420 2 2435 7368
Fax: +420 2 2435 7610
Web: dce.felk.cvut.cz/cak
E-mail: xdolejs@lab.felk.cvut.cz
Date: 14. Dezember 2001*

CAk
CENTRUM APLIKOVANÉ KYBERNETIKY

1. Introduction

The research community proposed several type of multicast communication over the Internet. The multicast communication is implemented in Internet protocol IPv4 and IPv6. This implemented version of multicast reserve a set of addresses for multicast groups. The network routers are able replicate packets and deliver them for members of group, it is suitable for sending packets in one direction e.g. radio and TV.

But this mechanism isn't suitable for bi-directional communication; the back channel is needed for a message acknowledgement and retransmission. The other important thing is the possibility to send a message to the others members of multicast group e.g. in Video conferencing.

To make change in this mechanism is too difficult and it takes long time, due to changes in the network routers and in the end systems. This situation led research community to propose mechanisms and protocols to build overlay spanning tree protocol.

This report is divided into six chapters. Tree Building Control Protocol is explained in the chapter 2 and it is passed from [1]. The TBCP state machine is described in the chapter 3. The modelling analysis is made in the chapter 4 and the implementation of TBCP state machine in OPNET® Modeler is given in the chapter 5.

2. Tree building control protocol (TBCP)

TBCP is a generic Tree Building Control Protocol designed to build overlay spanning trees among participants of a multicast session, without any specific help from the network routers. TBCP therefore falls into the general category of protocols and mechanisms often referred to as Application-Level Multicasting. TBCP is a distributed protocol that operates with partial knowledge of the group membership and restricted network topology information. The decision, where will be new member placed depends on distributed algorithm.

2.1. TBCP Join Procedure

In TBCP, the root of the spanning tree (e.g. main sender to the group) is used as a rendezvous point for the associated TBCP tree, that is new nodes “join” the tree at the root. Hence a TBCP tree can be identified by the (S,SP) pair, where S is the IP address of the TBCP tree root, and SP the port number used by the root for TBCP signalling operations. This information is the only advertised information needed to join the TBCP tree. Each TBCP entity (including the tree root) fixes the maximum number of “children” it is willing to accommodate in the spanning tree. This value, called the fanout of the entity, allows the entity to control the load of traffic flowing on the TBCP tree that it will handle. The TBCP Join procedure is a “recursive” mechanism and works as follows:

A newcomer N contacts a candidate parent P with a HELLO message, starting with the tree root S (Fig. 1(a)).

P sends back to N the list of its existing children C_i in a HELLO ACK message (Fig. 1(b)), starts a timer T_0 and wait for a JOIN message from N.

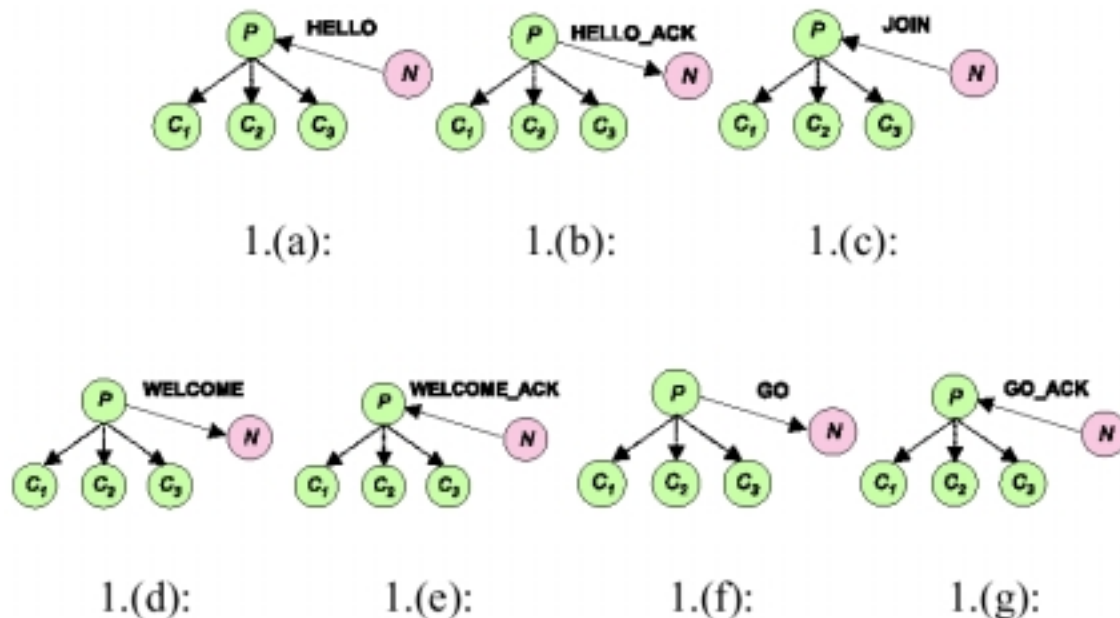


Fig. 1 TBCP Join procedure messages

This timer is needed because, for consistency reasons, P cannot accept any new HELLO message until it has finished dealing with the current newcomer.

N estimates its “distance” (i.e. takes measurement samples) from P and all C_i s and sends this information to P in a JOIN message (Fig. 1(c)). Note that if P has not received this JOIN message within its timer T_0 , P sends a RESET message to N, meaning that N needs to restart the procedure from stage 1.

P finds a place for N, by evaluating all possible “local” configurations having P as parent and involving $C_i \cup N$ (see Fig. 2). A score function is used to evaluate “how good” each configuration is, based on the distance estimates among P, N and the C_i s.

Depending on which configuration P has chosen, the following occurs:

if N is accepted as a child of P, P sends a WELCOME message to N, which N acknowledges immediately (Fig. 1.(d) and Fig. 1.(e)). The join procedure is then completed for N.

(b) if either N or any of P 's children (say C_j) is to be redirected to one of P 's children (say C_k), that node is sent a $GO(C_k)$ message (Fig. 1(f)) and starts a join procedure (stage 1) with C_k assuming the role of P and C_j the one of N . When N receives such a message, it acknowledges this immediately with a GO_ACK message (Fig. 1.(g)). However, in order not to disrupt the flow of data for an already established node, C_j is given a time to find its new place in the tree.

Notice that a Join procedure is not exclusively performed by TBCP entities that have not joined the tree yet. But, even a TBCP Entity that has already joined the tree may be forced to start a Join procedure, to find a new place in the tree. It should also be noted that the algorithm always finishes, as GO messages always send a TBCP entity (and the associated TBCP subtree whose root is the corresponding TBCP entity) down the TBCP tree.

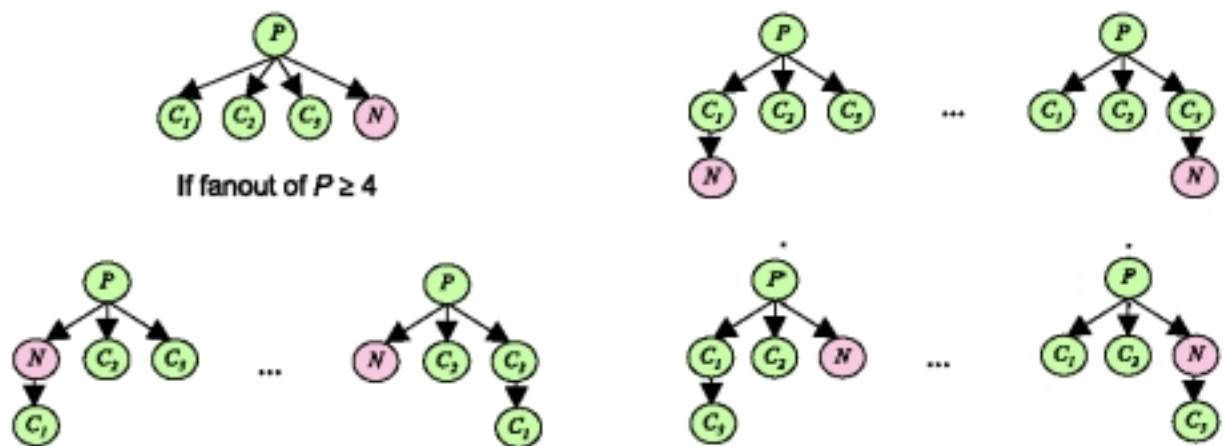


Fig. 2 Local configuration tested.

3. TBCP state machine

The finite state machine of TBCP protocol (Fig. 3) was updated to the ver. 1.1. The state could be divided into three main parts called:

The newcomer state machine: new request for join to the TBCP tree.

State:

- SEND_HELLO to P – the newcomer received request from API to join the tree, it sends a HELLO message to the candidate parent (source) node
- GET_HELLO_ACK – the newcomer received a HELLO_ACK message with list of children from candidate parent, the newcomer makes distance measurement and sends a JOIN message to the candidate parent
- GO_1 – the newcomer received a GO message, which contains new candidate parent, and sends a GO_ACK message back to the last candidate parent
- SEND_WELCOME_ACK_1 – the newcomer received a WELCOME message and it saves candidate parent as its parent, the newcomer sends a WELCOME_ACK message to the parent node

The parent state machine: accepts a newcomer or redirected node to the TBCP tree.

State:

- SEND_HALLO_ACK – the candidate parent node received a HELLO message, it sends HELLO_ACK message with list of its children
- CALCULATE – the candidate parent received a JOIN message with distance measurement, it calculates score function (SF) and makes decision, it sends messages depending on the result of the score function:
 - SF=1 - the newcomer goes direct under candidate parent
 - it sends a WELCOME message to the newcomer
 - SF=2 - the newcomer is redirected under a child
 - it sends a GO to Y message to the newcomer
 - SF=3 - the newcomer goes direct under parent node and one child of parent list goes under the newcomer
 - it sends a GO to the newcomer message to a child and a WELCOME to the newcomer
- WELCOME – the candidate parent node received a WELCOME_ACK message from newcomer, it saves newcomer as a new child and it goes to the state WAIT
- GO_ACK – the candidate parent received a GO_ACK message from newcomer or a redirected node and if it was the redirected node, the candidate parent discards its from the list of children and it goes to the state WAIT
- ADD_X_1 and ADD_X_2 - the candidate parent received a WELCOME_ACK message, it saves the newcomer as the new children
- REMOVE_Y_2 and REMOVE_Y_1 – the timer T1 expired or the candidate parent received GO_ACK from Y, it removes Y from the list of children

The redirect machine: redirect one child under other node.

State:

- SEND_HELLO to Y – a child received a GO message from its parent, it sends a HELLO message to the new candidate parent node
- GET_HELLO_ACK_2 – the child received a HELLO_ACK message with list of children from candidate parent, the child makes distance measurement and sends a JOIN message to the candidate parent
- GO_2 – the child received a GO message, which contains new candidate parent
- SEND_WELCOME_ACK_2 – the child received a WELCOME message and it saves candidate parent as its parent, the child sends a WELCOME_ACK message to the parent node and a GO_ACK message to the last parent node

The common state for all parts is:

- INIT - variables and statistics initialisation
- WAIT – wait for event, no program code

State Machine

of the Tree Building Control Protocol (TBCP)

ver. 1.1

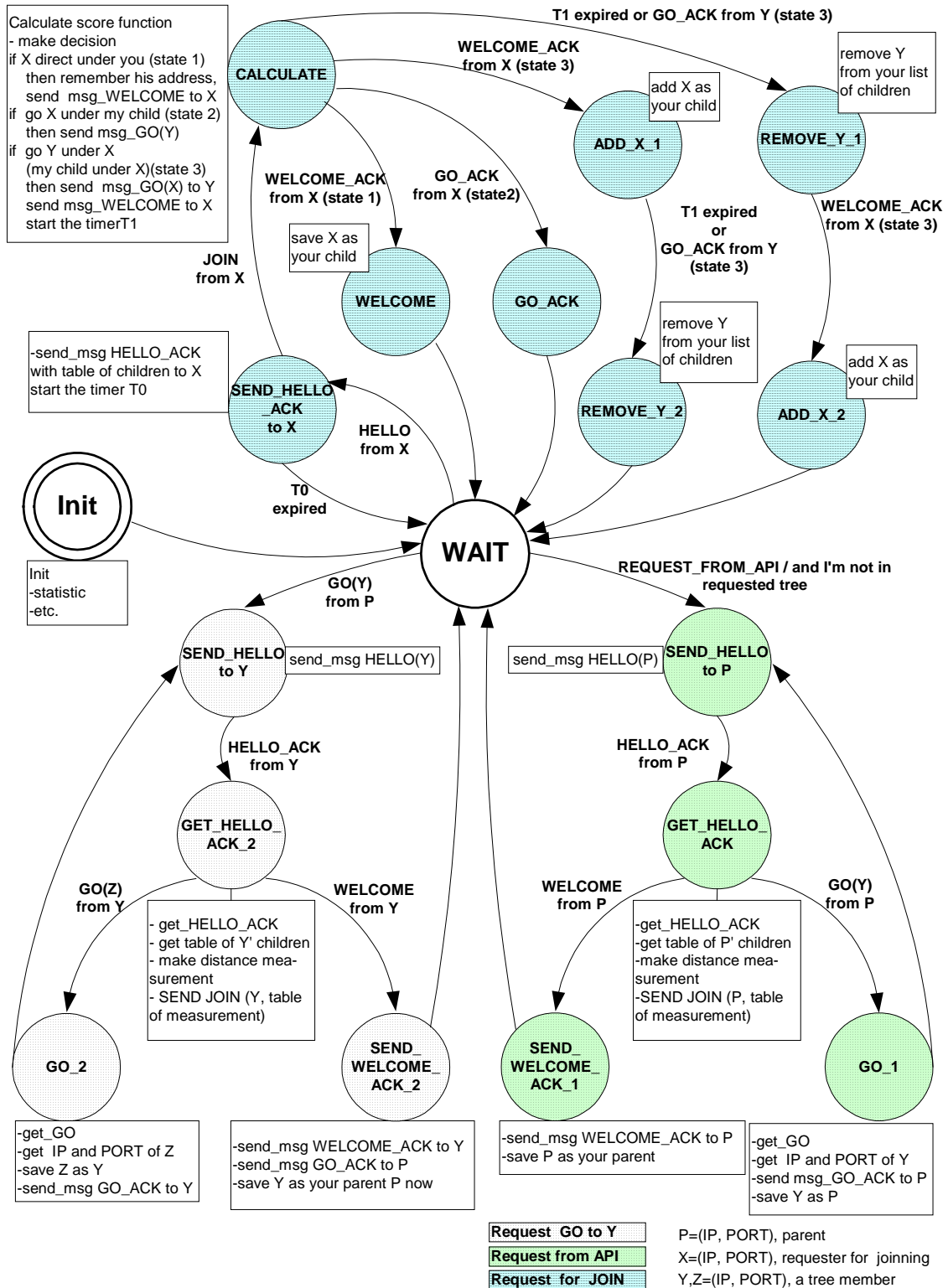


Fig. 3 The TBCP state machine

4. Modelling analysis

The simulation and implementation phase could be divided into the three following phases:

4.1. Logical modelling

This phase focuses on simulation of logical behaviour of TBCP protocol. It means to make simulation without network protocol such as TCP/IP etc. This phase could be divided into further steps:

- to construct state machine - described in [2]
- to update state machine up to the last protocol version
- to check algorithm correctness e.g. deadlock
- to make simulation for different amount of nodes and size of fanout

4.2. Functional modelling

The aim of this phase is to simulate whole TBCP protocol with exploitation of OPNET Modeler's models library such as TCP/IP protocol, routers and links. It means that the simulation should be close to the real implementation as much as possible.

- model debugging simulation with basic set of nodes (e.g. 4 nodes)
- simulation under worst case conditions (e.g. 100 nodes, heavy load,..)
- simulation of real configuration based on the experiment parameters

4.3. Discussion of results

After the phase 1 the optimality of the spanning tree generated by TBCP protocol will be evaluated. The TBCP protocol is distributed algorithms without complete knowledge of the network topology and parameters, therefore the optimal spanning tree couldn't be achieved in all instances. The interest is to find how far is this algorithm from the optimum and what is the optimum. Is it a minimal spanning tree or load of each links and routers?

Criteria for score function:

- RTT (Round Trip Time)
- maximal distance (diameter, time delivery delay) between root and leaf (between child and parent) , number of hops - depends on the depth of the tree
- preferences of the some links or nodes

Note: the minimal spanning tree doesn't have influence to the mean distance between nodes and root, because the minimal spanning tree minimizes the total sum of distances between neighbourhoods.

5. Implementation of TBCP state machine

Some correction of TBCP state machine described in [2] had to be done and logical machine of TBCP protocol based on (Fig. 3) was implemented in OPNET Modeler 8.0.C (Fig. 4) as a new process model. The simple score function is used according to chapter 4.1.

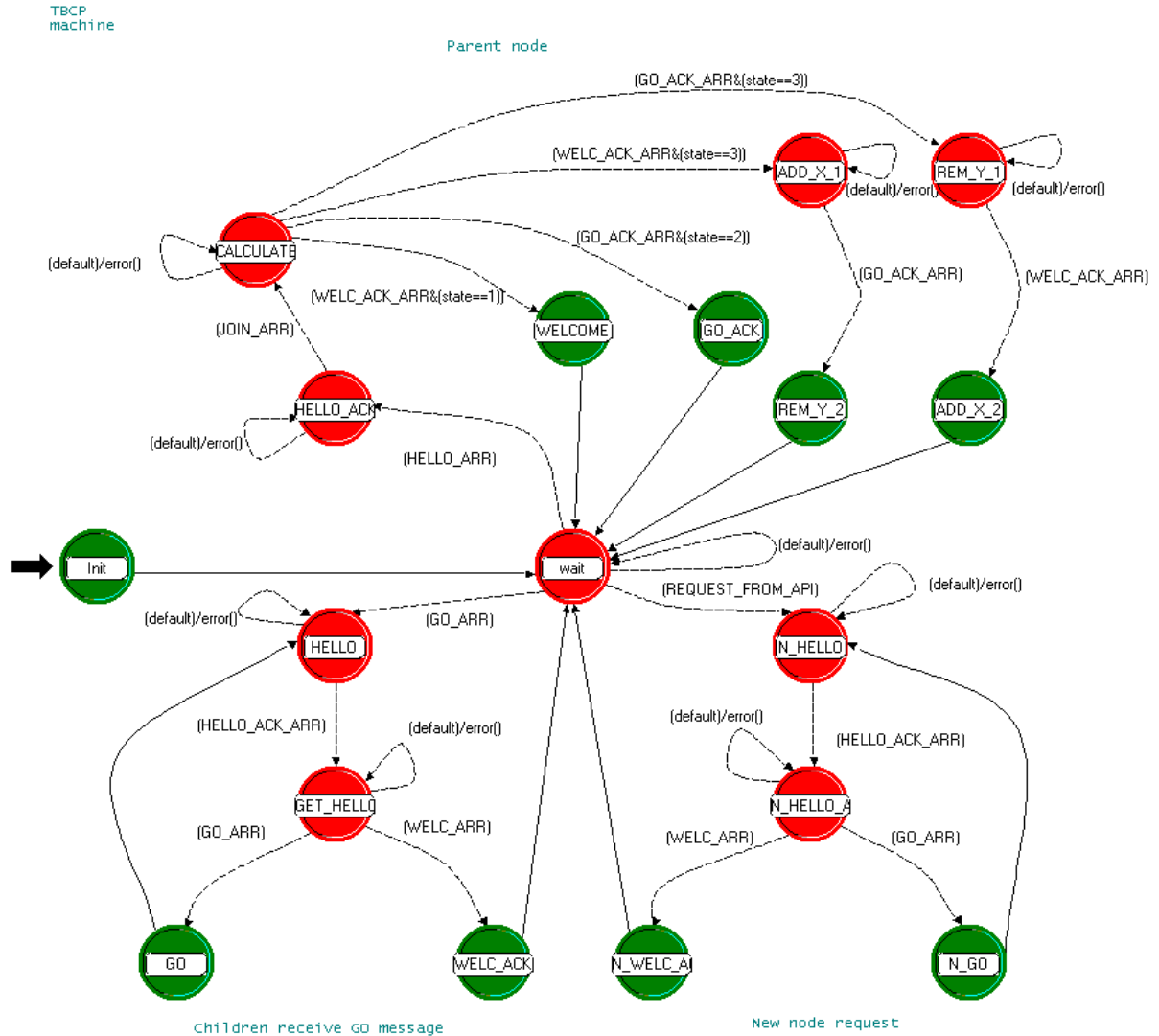


Fig. 4 Implemented TBCP state machine

6. Further work

The simulation with different score functions and different network's configuration will be designed. The logical correctness of TBCP protocol will be tested and discussed. The most important is analysis of the constructed TBCP tree as a result of simulation for different number of node and RTT (Round-Tripp-Time) and comparison with real configuration of network.

Literature:

- [1] Mathy, L., Canonico, R. and Hutchison, D. , An Overlay Tree Building control Protocol, Lancaster University, England, 2001
- [2] Dolejš, O., Report on the GCAP meeting in Toulouse concerning tasks for Czech Technical University in Prague as a new member of GCAP, Toulouse, July 2001