# Integration of Rules and Linear Program Models in Network Traffic Management

**G. Martini, C. Moiso, M. Porta**

**CSELT**
**via Reiss Romoli 274, 10148 Torino (Italy)**

**Abstract**

In the last years, several attempts have been made to apply AIP techniques to develop systems assisting the network supervisors in selecting the controls to handle network difficulties. Some of the approaches adopted are based on rule-based expert systems and operative research methods: all these attempts have their advantages and drawbacks.
This paper describes an approach based on the integration of symbolic rules and linear program models, based on the Constraint Logic Programming (CLP) paradigm, in order to merge the advantages and overcome the problems.
By using this approach, we have developed an experimental decision support system to assist the network supervisors in the selection of the expansive controls (e.g. TAR's, Temporary Alternative Routes) to improve the network performance during serious congestion states.
The system is under testing in a simulated environment.

## 1. Introduction

The problem we tackle in this paper is that of traffic management in a telephone network, when serious congestion states occur caused by external factors. In these situations, the network, planned for normal busy-period loads, is unable to carry offered traffic (due to either abnormal traffic patterns or network elements failures) and the service performance decreases both for the repeated attempts phenomenon and for the spread of the congested area.

These events must be detected and controlled at the traffic management centre. This centre collects current statistics on the network state, by using which abnormal behaviours are signaled, classified and controlled through the application of expansive and protective actions. The former set of controls allows to introduce additional routes, called Temporary Alternative Routing (TAR), to the routing table of a relation, while the latter one filters at the origin node a part of the call attempts of a relation. It is important that the actions applied to control a congestion are sufficient to reduce

it without producing a congestion in other parts of the network or blocking an excessive amount of calls.

The selection of controls to increase network behaviour is a hard work for the network managers: they must take into account several aspects (the available resources, the current load, the call routing, the network topology,...) and different requirements, according to both the telephone company strategy and the CCITT recommendations. In addition to these issues, the changes in the topology and the trunk capacities, performed during the periodical updates of the network, can invalid the experience acquired by the traffic managers.

## 2. Advance InformationProcessing techniques for traffic control

In the last years, several attempts have been made to apply AIP techniques in the development of systems that assist the network supervisors in selecting the correct controls to handle network difficulties.

The different technologies are characterized by the kind of the adopted knowledge acquisition and representation:

1) rule-based formalism: several expert systems [1,2] were developed through the synthesis of the knowledge provided by the experts into if-then symbolic rules, which are processed by an inference engine;
2) neural network approach: the controls to be performed are learned through training algorithm on a sensible set of examples [3];
3) mathematical models: the network behaviour and the action effects are described through a system of equations/disequations, solved through operative research techniques (e.g. the simplex method) [4].

All these approaches have their advantages and drawbacks. One of the disadvantages in 1) is the difficulty to synthesize knowledge from human experts, by using only if-then rules; moreover it is hard to include optimization criteria in pure rule-based reasoning. In 2) it is difficult to produce from examples on one network rules valid for a class of different network topologies.

The last approach seems to be quite promising because the behaviour of a generic network is formalized through a mathematical model, that can be easily instantiated to a particular network. But such a model is not able to represent other aspects of the problem, such as constraints on the kind and/or the number of the actions to be performed or principles ruling the control activities. In fact, it is quite difficult to code as numerical constraints all the conditions and the rules

involved into the definition of a strategy to control network congestions.

## 3. An integrated approach

We adopted an integrated approach that is based both on rules and mathematical models: in this way we mix the so-called deep knowledge of the system (in our case formalized through a mathematical model that represents the behaviour of the network with some approximation) with the so-called shallow knowledge (expressed through a set of symbolic rules, that determine the strategy of the controls according to some principles defined by the domain experts).

In addition to these two forms of knowledge, the description of the problem is completed by a criterion used to choose the "best" solution among those that satisfy both the numerical and the symbolic constraints. The criterion is introduced as a cost function to be optimized (i.e. minimized/maximized): different functions single out different objectives of the telephone company, such as the carried traffic or the number of rejected calls.

The advantage of the integration of the two knowledge representation techniques is that each piece of knowledge in the system can be coded in the most suitable way: knowledge about the behaviour of the network and quantitative conditions can be easily described through a mathematical model, while the heuristic rules synthesized from the domain experts can be expressed through symbolic logic formulae. In this way we avoid some cryptic (and often incomplete) coding of one form of knowledge in the other one. Moreover changes and extensions of the knowledge in the system can be performed very quickly, because there is an almost one-to-one (declarative) correspondence between the knowledge and its coding.

There are some limits in the class of mathematical models we can use to model the network behaviour. In fact, we have to find a compromise between the accuracy of the model and its computational characteristics.

It is well-known that the network behaviour has non linear models (see, for instance, the B-Erlang formula). Unfortunately there are no generic tools to solve non-linear systems and, then, to perform optimizations w.r.t. them. Generic tools are instead available for linear systems (of equations/disequations), both to solve them and to optimize linear functions w.r.t. them: an example of such tools is the simplex method.

### 3.1 Constraint logic programming and the language VEL

In the last years, a new computation paradigm, the constraint logic programming (CLP), has been developed [5], as an extension of logic programming, suitable for the representation and the computation of both numerical and symbolic knowledge. Several extensions of Prolog, the most important and used logic programming language, has been designed according to the new paradigm: the search-based symbolic computation mechanism of Prolog (based on the resolution inference rule) is enriched with the possibility of dealing with numerical conditions (in the general case, conditions in an algebra). When in the computation of a branch of the search tree a constraints must be resolved, it is added to the set of those previously met in the same branch:

- a special solver algorithm is invoked to check the consistency of the new set of (numerical) constraints (and possibly to compute its solution);
- if the solver detects an inconsistency the current branch fails, and the next one is tried (according to the Prolog operational semantics).

The CLP paradigm is parametric w.r.t the domain of constraints. It can be instantiated, according to the application requirements, with the suitable kinds of constraints (e.g. constraints on real number, on integer number, or on boolean values): each constraint domain requires the development of a constraint solver algorithm to be invoked during the computation.

The CLP paradigm is not just a simple mix of numerical and symbolic computations; in fact it has several additional advantages:

- the two forms of conditions can be present in the same rule: there is a real integration of the two form of knowledge and not just a coupling of them;
- the numerical constraints can be added in an incremental way: this is in contrast with the operative research tools that require that the set of constraints is provided as a whole;
- disjunctive numerical constraints can be handled: the operative research tools are not able to cope with this kind of constraints in an efficient way.

All these properties have to be efficiently supported by the adopted solver algorithms: they must be incremental and integrated with the mechanism implementing the Prolog search-based computations.

The following program shows up the declarative way of programming in CLP and some of its properties: it checks the condition cond(X,Y,D) that holds when there is a path in a graph from X to Y shorter than D (we assume to consider constraints on real numbers, introduced between curly brackets):

```
cond(X,Y,D) :- {S=<D},path(X,Y,S).
        cond(X,Y,D) is true if
            there is a path from X to Y of length S, such that S<D

path(A,B,S) :- edge(A,B,S).
        there is a path from A to B of length S if
            there is an edge from A to B with length S.
path(A,B,S) :- {S = S1+S2},edge(A,C,S1),path(C,B,S2).
        there is a path from A to B of length S if
            A is connected to a node C with an edge long S1 and
            there is a path from C to B long S2, with S=S1+S2.

edge(turin,paris,900).
edge(paris,london,500).
.......
        a database of facts describing the edges in the graph.
```

We have designed and implemented a member of the CLP language family, VEL (Vincoli e Logica), that extends Prolog with the facility to cope with constraints on rational numbers [6]:

- VEL handles linear equations and disequations on rational numbers;
- VEL has a rich set of primitives to maximize/minimize a function w.r.t. a set of constraints.

The optimization primitives, which are the peculiarity of VEL w.r.t. analogous proposals, were conceived to cope, in an efficient way, with disjunctive constraints and with the incremental addition of constraints: therefore VEL is suited to compute optimal solutions in a search space.

One of the application fields of VEL we had in mind when we designed it is the development of near real-time expert systems. Therefore we paid attention to the efficiency of its implementation.

VEL programs are compiled into an extension of the abstract machine for Prolog, where the elaboration of the numerical constraints is tightly integrated with the implementation of the Prolog functionalities. The solvers and the optimization algorithms are implemented in C.

## 4. A traffic control problem

We have developed in VEL an experimental decision support system to assist the network supervisors in the selection of the expansive controls to improve the network performance during serious congestion states. The system is based on the approach, that integrates the rule-based and the model-based ones, described in the previous section.

The telephone network we considered is a hierarchical one, with two levels of nodes (terminal nodes and transit exchanges), with a hierarchical, step-by-step routing control.

The network behaviour is monitored by a supervisor system that provides a set of network parameters, updated at fixed time periods, and performs alarm detection functionalities. Some of the data used in the developed system are: trunk congestion alarms, the traffic offered to a relation, the trunk loss probabilities, an estimation of the resources available at each trunk in the following interval.

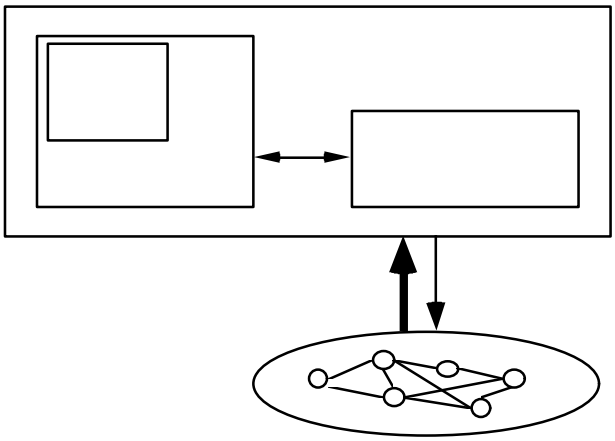
We assume that the exchanges in the network are able to perform the following control commands:

- selective tar command:

  tar n <i,$n_1$,...,$n_k$,j>

  its execution adds the path <i,$n_1$,...,$n_k$,j> (where $n_1$,...,$n_k$ are the k transit node of the expansive route) as the n-th route in the routing table of the relation (i,j); the tar command can have an additional parameter, the percentage of call attempts offered that can overflow on the expansive route.

The support system we developed is a component of the traffic control module implementing the strategy described in [7]. In the network supervisor system this module is placed between the monitor module (which collects data and detects alarms) and the traffic managers.



The support system recommends to the network managers the set of relations to be expanded and the actions to be applied in order to optimize the network performance. In order to keep the controls simple and reduce the effects on the other relations, an expansive control for (i,j) consists of the introduction in its routing table of a TAR, that diverges as soon as possible from the planned routes; moreover the expansive route has only one node (called the pivot node of the TAR) not present in the planned routing table.

The decision on the controls to be applied is made according to the principles of the strategy and the adopted cost function to be optimized.

## 5. The decision support system

The developed system has as input a list LEXP of relations that may be expanded and returns as output a sequence of tar commands on a subset of the relations in LEXP. The computation of the action sequences requires some further data on the network current status (routing tables, parameters, alarms,...) that are got from the monitor system.

The selection of the relations in LEXP to be expanded and the choice of the actions to be activated are based on different kinds of knowledge, and can be affected by the managers through a user(-friendly) interface.

The processing performed by the support system can be sketched in the following steps:

1) input of the list LEXP;
2) filtering of the list LEXP, by performing some correlation among the relations in it and the congestion state of their final trunks and transit exchanges;
3) assignment to each relation in the filtered LEXP of a control scheme, according to a set of rules that implement some principle of the control strategy (developed by the domain experts); each scheme is parametrized w.r.t. a set of possible TARs;
4) selection of the relations to be actually expanded and choice for each of them of the TAR to be performed, according to a linear model and a (cost) function to be optimize;
5) output of the sequence of actions according to the scheme of the selected relations, instantiated with the chosen TARs.

In the rest of this section we briefly analyse the main aspects of these steps.

The list LEXP (which is computed according to the strategy in [7]) contains the relations (i,j) that could suffer from congestion inside the toll network, but whose call attempts successfully routed to the destination node j have a high probability of reaching the user. These relations are characterised through the following conditions:

- unacceptable end-to-end blocking probability;
- good ASR (answer seizure rate) in the destination node.

Due to the variability of measurements because of the stochastic nature of the traffic, it is necessary to perform some correlations among different parameters in order to make safer

the detection of abnormal behaviours. As the considered network is hierarchical, the final trunks are good indices of the network behaviour; therefore LEXP is filtered to remove from it all the relations whose final trunks and exchanges are not congested, in order to focus the controls on those relations that actually suffer.

Step 3) assigns to each relation (i,j) in the filtered LEXP a control scheme and a set of possible pivots for the TAR (called pivot(i,j) in the following). A scheme is a template that describes the structure of the controls to be applied to a relation: it contains the following data, that specify the arguments for the tar command:

- the position in the routing table where to insert the TAR;
- the template of the TAR;
- the traffic offered to the TAR.

The template of the TAR is parametric w.r.t. the pivot node: it is the sequence of the nodes of the path, where the position of the pivot node is shown by the character *. The template of the TAR of the relation (i,j) can be instantiated with one of the nodes in pivot(i,j) to provide a possible TAR for (i,j): the actual pivot of the TAR is selected in the optimization phase in step 4).

The pivot(i,j) set is built according to the topology of the network and to the state of the trunks; if

<i,t,*,j>

is the template of the TAR in the scheme for the relation (i,j), then a transit node n is put in the pivot(i,j) set if the following conditions hold:

- n is a transit exchange that does not occur in the current routing table of (i,j);
- there exist the trunks t->n and n->j (called the pivot trunks);
- n, t->n and n->j are not congested;
- the trunks t->n and n->j have residual capacities available.

The residual capacities of a trunk t is an estimation of the circuits of t that will be free during the following time period.

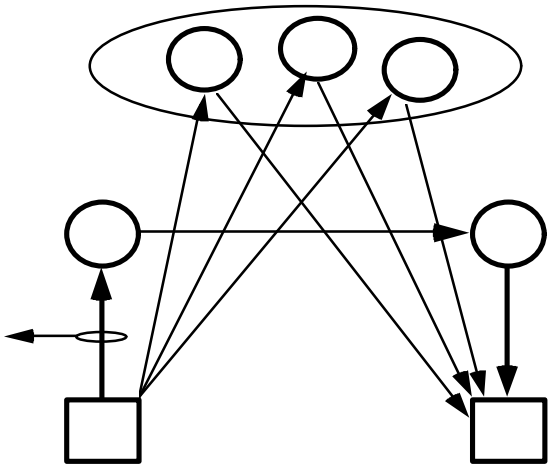
If pivot(i,j) is empty, the relation (i,j) is removed from the list of relations to be expanded.

The control scheme are assigned to the relations according to a set of rules. The assigned scheme depends on the following parameters:

- the current routing table of the relation;
- the congestion state of the final trunks and transit exchanges;

In some cases the assignment depends also on the available pivot nodes.

Let us consider the following portion of a hierarchical network, where I and J are two terminal nodes (connected only through the final route) and Ci and Cj are their respective transit exchanges.



An example of rule is:

if
1) in the routing table of the relation (I,J) there is only the final route
and
2) the trunk I->Ci is in congestion
then
the control scheme for (I,J) is:
- position of the TAR: 1;
- template of the TAR: <I,*,J>
- traffic offered to the TAR:
traffic offered to (I,J) * loss prob. of I->J

The set {h,k,y} is the set of nodes that satisfy the previous conditions for pivot(I,J).

These rules are coded by exploiting the logic features of VEL.

Step 4) determines the relations to be actually expanded and for each of these selects (from their pivot sets) the pivot node of the TAR.

These choices are driven by a model of the traffic flows and a function (to be maximized) that introduces a criterion of best solution, i.e. the solution that optimizes the network performance (according to the aspects covered by the adopted cost function definition).

The model is a linear one, where the trunks are traffic limiters, similar to that introduced in [4], but with disjunctive linear constraints. In the model, for each relation (i,j) to be expanded the following variables, denoting traffic flows, are introduced:

- $A'_{ij}$: the traffic carried by the TAR chosen for (i,j);
- $X_k(i,j)$, where $k \in$ pivot(i,j): the traffic carried by the possible TAR for (i,j) with k as pivot.

For each relation (i,j) there are the following constraints:

- $A'_{ij} \geq 0$ and $A'_{ij} \leq A_{ij}$
  the traffic carried by the TAR of (i,j) can not exceed $A_{ij}$, the traffic offered to the TAR;
- $A'_{ij} = \Sigma_{k \in pivot(i,j)} X_k(i,j)$.

For each pivot trunk we introduce a constraint, whose meaning is: the sum of the (expanded) traffic routed through the trunk can not exceed the amount of the residual resources of the trunk. This set of constraints limits the amount of the (expanded) traffic that a trunk can carry without damaging the other relations.

The function to be maximized is the total expanded traffic: $\Sigma_{(i,j)} A'_{ij}$.

At the moment, we assume that for each relation there is at most one TAR. The program tries (in an intelligent and efficient way) all the possible pivot nodes for each relation (including the possibility of selecting none of them). This is modelled through a set of disjunctive constraints.

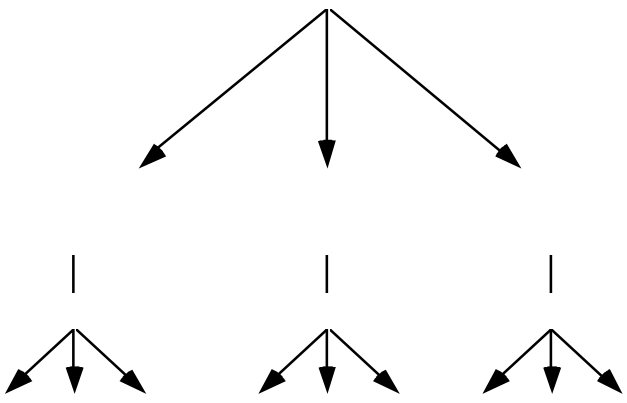
The selection of k as pivot for the relation (i,j) is modelled by the following set of constraints:

- $X_h(i,j) = 0$, where $h \in$ pivot(i,j) and h_k
  only the route through k can carry the traffic of (i,j) to be expanded;
- $X_k(i,j) \geq p * A_{ij}$
  the TAR through k must carry at least a percentage p of the offered traffic.

The decision of not expanding a relation (i,j) is modelled by:

- $X_h(i,j) = 0$, for each $h \in$ pivot(i,j)

The constraints are disjunctive because different choices of the pivot (explored in different branches of the search space) are modelled through different sets of constraints.



The numerical constraints are integrated with additional conditions, such as a limit on the number of TARs to be activated.

The program produces the solution that maximizes the cost function. In addition for each selected TAR the model computes the amount of traffic $A'_{ij}$ that it can carry without damaging the other relations. There may be some relations for which no pivot is chosen: in fact, we have to perform a global optimization, whose result (i.e. the best solution) is not just the sum of the best TAR of each relation.

This step is implemented by exploiting the features of VEL to perform in efficient way optimizations in a search space, constrained by both numerical and symbolic conditions.

The control scheme associated to the relations are combined with the result of this step to produce the set of actions to be activated. The actions are performed only on those relations for which a pivot is chosen; the selected pivot and the amount of the carried traffic are used to build the parameters of the tar commands:

- the pivot node instantiates the template of the TAR to form the path of the route;
- the percentage of the TAR is computed from the amount of the carried traffic.

The support system has a user interface to interact with the supervisor. The supervisors can require different solutions computed according to different sets of parameters (e.g. the maximum number of TARs), compare them and, then, select the one that they prefer.

## 6. Conclusions

The described support system is currently under testing in a simulated environment [8], on a hierarchical network of 16 terminal nodes and 4 transit exchanges, with an high degree of interconnections.

After a first set of tests the following issues can be argued on the approach and the strategy:

- by considering the tests on the current network and some hand-made examples corresponding to bigger networks, the system has an execution time suitable to be used as a near real-time decision tool;
- the LEXP list construction and the correlation phase have a crucial role in order to detect the set of relations that actually suffer: a careful tuning of the involved thresholds is required;
- further study is needed to cope with the relations that can not be expanded either at all or in a satisfactory way;

- the CLP paradigm allowed both a rapid development of the system and a quick implementation of extensions and changes of the strategy, the mathematical model and the heuristics used to shrink the search space (without loosing the optimal solution).

## 7. References

[1] **S. Jimenez**, The application of ES to Nework Traffic Management in Telecom Australia, in Telecommunication Journal of Australia, vol. 38, n. 1 (1988), 25-33.

[2] **M. Georgeff, A. Rao**, Intelligent real-time network management, in Proc. 10th Int. Workshop on Expert Systems and their Applications (1990), 87-101.

[3] **B. Silver**, Netman: a learning network traffic controller, GTE Labs Report.

[4] **R. Warfield, D. McMillan**, A linear program model for automation of network management, in IEEE J. on Selected Areas in Comm. 6, n. 4 (1988), 742-750.

[5] **L. VanHentenryck**, Constraint satisfaction in logic programming (MIT Press, 1989).

[6] **C. Moiso, M. Porta**, VEL: a constraint logic language for the integration of symbolic and numerical knowledge, in Proc. 6th Italian Conf. on Logic Programming (1991), 365-375.

[7] **M. Buttò, G. Giacobbo Scavo**, Network management policies: which aims and how to pursue them, in Proc. Network Management and Control Workshop (1989).

[8] **F. Malabocchia, et al.**, A real-time simulation tool for network management applications, in Proc. SimTec'91 (1991).