Network planning with constraint programming

Laurent Jeannin[1], Simon de Givry[2]

1. THALES Research & Technology, Domaine de Corbeville, 91404 Orsay cedex
laurent.jeannin@thalesgroup.fr

2. INRA, BP 27, 31326 CASTANET-TOLOSAN cedex
degivry@toulouse.inra.fr

Abstract

This paper presents a network planning application developed at THALES Research and Technology. The problem is to determine the successive evolutions to make in a GSM network so that all forecasted demands are satisfied and the global cost of investment and infrastructures maintenance is minimized. The application was modeled with a constraint solver over finite domains. The resolution algorithm was designed according to the architecture defined in EOLE project and developed with the ToOLS © (Templates of On-Line Search) library. In this framework, a search algorithm is the conjunction of four distinct components: (1) a set of heuristics to order choices, (2) a search scheme which represents a search tree, (3) a search strategy applied to a search scheme to control its complexity and (4) a temporal strategy allowing the management of time for the hybrid search algorithms. We illustrate the relevance of such a library with a quick prototyping and a resolution algorithms comparison (partial searches and hybrid algorithms).

Keywords: constraint programming, network planning, large neighborhood search.

1. Introduction

This paper presents a network planning application developed at THALES Research and Technology with the EOLE[1] (Environnement d'Optimisation temps réel en LignE dédié télécoms) project framework. Network planning follows a complex process, where many parameters of very different nature (economic, political, strategic) are to be taken into account. Solving the whole problem seems difficult. Our goal is to determine and to resolve a relevant sub-problem to help the network designer in his search for a good solution. This kind of use relies on a strong interaction between the designer and the tool which must have a short time contract. This is a simulation approach: the designer gradually refines the model parameters according to the resolution algorithm results.

The problem is to determine the successive evolutions to bring to a GSM network so that all forecasted demands are satisfied and the global cost of investment and infrastructures maintenance is minimized. Demands are indivisible (e.g.: each demand must be routed along a single path). This mono-routing problem with linear side constraints is NP-complete [Bertsekas 1998].

The application was modeled with a constraint solver over finite domains. To obtain good solutions in limited time, we used a framework dedicated to on-line optimization. We elaborated a partial search algorithm mixed with a large neighborhood search.

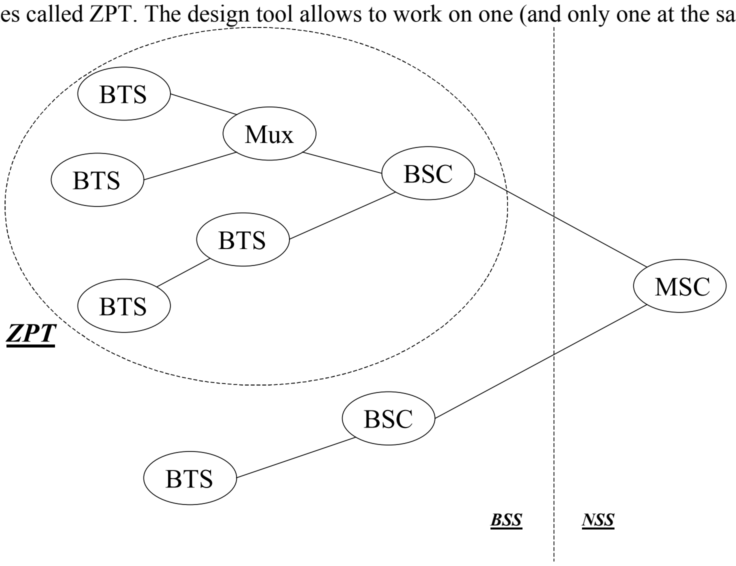
Our problem is a temporal extension of the one presented in [Le Pape & al. 2002] which proposes a comparison of various techniques (constraint programming, mixed integer programming, column generation) to resolve various variants of a network design problem, and focuses on robustness. Acutally, we solve a network design problem for several network states. Moreover we introduce recurrent costs in the objective function in order to have more realistic cost model. [Perron 2002] use the constraint programming to resolve the problem described in [Le Pape & al. 2002].

[1] This work was partially funded by RNRT (Réseau National de Recherche en Télécommunications)

The resolution phase is based on a large neighborhood search. In [Lauvergne and al 2002] a mono-routing problem is solved with an approach mixing a shortest path algorithm, constraint propagation and repair principles.
In the second section, we present the network planning problem. Section 3 deals with the constraint model. The resolution method and the search algorithms are described in the fourth section. We end with a comment on our experiments (section 5).

2. The network planning problem

A GSM network is divided into two subsystems: the Network Subsystem (NSS) and the Base station Subsystem (BSS). Our design tools works on BSS. BSS consists of mulitplexers (MUX), base station controllers (BSCs) and base transceiver stations (BTSs). The BSC provides physical links between the MSC and BTSs. The BTSs establish the connection of mobile stations in the cellular network . It basically consists of radio emitters and receivers, and antennae. These equipments are interconnected with different kind of connections: proprietary connections (hertzian bundles and optical fibres) and rented connections.
France is divided into 6 regions, which count each several hundreds BSCs and several thousand BTSs. Every region is subdivided into zones called ZPT. The design tool allows to work on one (and only one at the same time) ZPT.



The design tool allows to estimate the evolutions to be made in the network to answer new demands. A demand indicates a traffic evolution forecasted between 2 nodes of the network, (i.e. between a BTS and a BSC. A demand must be normally defined in two directions (the volume of traffic is not symmetric), but we suppose here that the debit is the same in 2 directions. A demand is thus defined by a BTS, a debit, a starting date and an ending date. We try then to determine the successive evolutions to make in the network so that all demands are satisfied and the global cost of investment and infrastructures maintenance is minimized. We try to establish roads as short as possible (in number of crossed equipments) for the demands (for reliability reasons).

3. Constraint model

The problem consists basically of the following sub-problems:
– A routing problem: choice of routes verifying links and debits capacities according to demands requirement;
– A configuration problem: choice of equipments and connections type and capacity;
– A planning problem: choice of evolution dates according to the fixed costs and the recurrent cost.

The network is represented by a graph. Vertices represent the real or potential sites; and edges the real or potential links. Equipments (links and sites) are characterized by an initial capacity, extension possibilities, creation and extension costs (fixed costs) and a maintenance cost (recurrent cost). Demands are characterized by an origin, a destination, a debit, a starting date and an ending date.

The planning problem is decomposed into several stages. Every stage corresponds to the state of the network in a given date and the list of demands to provide in this date. Decision variables are associated to describe the state of the network at each stage.

These sub-problems are handled by a global constraint model. Constraint Programming (CP) [Hentenryck 1989] is based on the cooperation of two processes: a thinking process that maintains local consistency and a guessing process that divides the problem into sub-problems by generating hypothesis.
– Local consistency is achieved by filtering algorithms which remove combinations of values that cannot appear in the solution under construction. This process is also called constraint propagation.
– Decomposition is based on committing choices to divide the problem into sub-problems until they get simple enough to be solved in a straightforward way. It is usually implemented by a tree-search algorithm. For optimization problems the algorithm is extended to a so-called branch-and-bound algorithm which principle is to avoid to search in sub-trees for which a proof has been made that no better solution can appear.

A constraint model has modularity properties, i.e. adding/removing a constraint is easy, which enables an incremental development process, reducing the development time and effort. CP solvers provide efficient algorithms through the use of global constraints. The declarative nature of CP enables the programmer to focus on the application requirements rather than on debugging low-level programming errors.

The application was modeled with Eclair© [Laburthe & al. 1998], the constraint solver over finite domains developed at THALES Research and Technology.

4. Resolution

The resolution algorithm is based on the architecture defined in EOLE [EOLE 2001] and developed with the ToOLS© library [de Givry&Jeannin 2003]. ToOLS (Templates Of On-Line Search) is a framework dedicated to the design of search algorithms. This framework provides an abstraction level which facilitates the design of customized search algorithms. Time management facilities are also offered in order to respect a time constraint. ToOLS offers a library of predefined search algorithms based on template of search. A *template of search* defines an abstract component of a search algorithm that can be reused to speed up the development process of customized partial search algorithms. This framework makes it easier to try new combinations of search limits and new temporal strategies. A search algorithm is expressed in ToOLS as the conjunction of four distinct components:
– A **set of heuristics** to rank every choice;
– A set of primitives to express a **search scheme** independent of any time limit; it is composed by predefined choice points and combinations of choice points as in the OPL language [Hentenryck99];
– A set of primitives to express the search limits that depend on the current node, the current path or the current sub-tree; the resulting parameterized search algorithm controls the size of the explored search tree defined by one search scheme;
– A **temporal strategy** defined by a hybridization scheme, i.e. a cooperation of several parameterized searches, dealing with time allocation and selecting the tuning strategy of the parameters (static tuning, iterative tuning or adaptive tuning).

4.1. Search scheme

A search scheme defines a structured view of a search space. It is implemented by defining an ordered set of choice-points. Each choice-point refers to a sub-set of variables presented in the CP model. A combination of choice-points defines a complete search-space.
We built a search scheme in two steps:
– Determine the routing of all demands by labeling,
– Deduct the required capacities of equipments to support this routing (The capacities of equipments are computed by constraint propagation).
This strategy can be quickly implemented thanks to primitives available in ToOLS.

4.2. Exploration strategy

An exploration strategy is applied on a search-space defined by a search scheme. Exploration strategies try to solve a problem by doing an efficient exploration of its search space. It describes which parts of a search will be visited and in what order. The goal of an exploration strategy is to focus the search on some promising parts of a search space, loosing completeness, but having in most cases better results in limited time.

ToOLS contains a search algorithms library easily applicable to a search scheme. It is also possible to define its own algorithms thanks to dedicated primitives. We tested the *CREDIT* partial search [Beldiceanu & al. 1998]: it limits choices at the bottom of the tree by granting a credit amount to nodes, to change the first choices made at the top of the tree.

4.3. Temporal strategy

A temporal strategy defines a search algorithm based on several searches and expresses the way of combining them. We applied a hybrid strategy based on local search with large neighborhood [Loudni&Boizumault 2001], using the technique of shuffling to define the neighborhood [Applegate&Cook 1991]. A part of the routing is undone, and then reconstructed in an alternative way.

The shuffling search looks for an initial solution. The neighborhood is defined by a subset of decisions variables. All variables that are not in this subset (called the frozen variables) are assigned to their corresponding value saved in the last solution found so far. The initial neighborhood is composed of one variable. The hybrid search method tries different searches in a cyclical way. If no solution has been found in the current neighborhood, then the number of variables put in the neighborhood is increased by one, and the same cyclical process starts again with this new neighborhood.

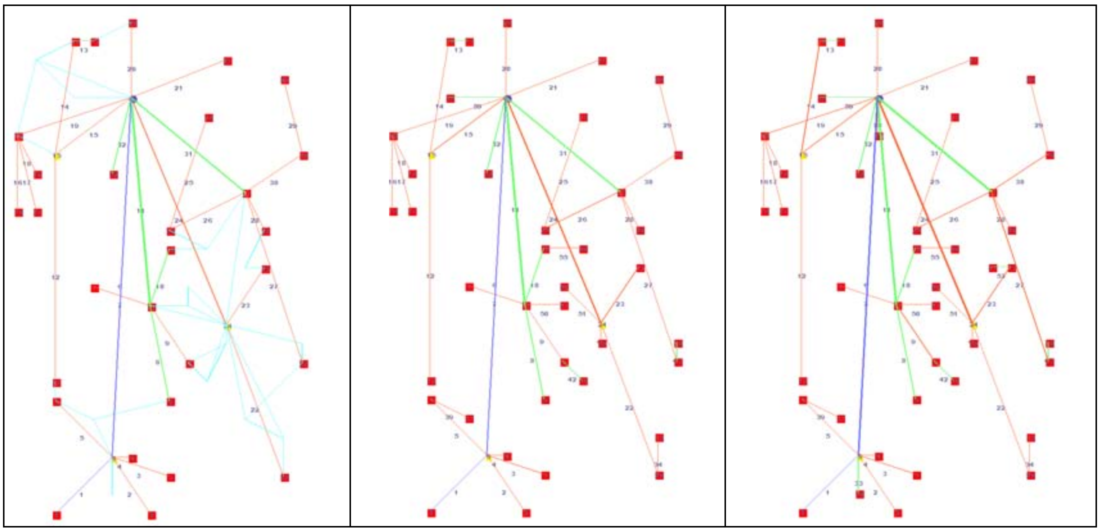
We distinguish two ways to define the neighborhood: on any set variables of routing (*shuffle1*) and on a set of variable which defined a path for one or several demands (*shuffle2*).

5. Experiments

In the instance of the problem considered for the experiments, the planning horizon is fixed to 8 quarters, corresponding to 2 years (a step of time = 1 quarter) and the network consists of:

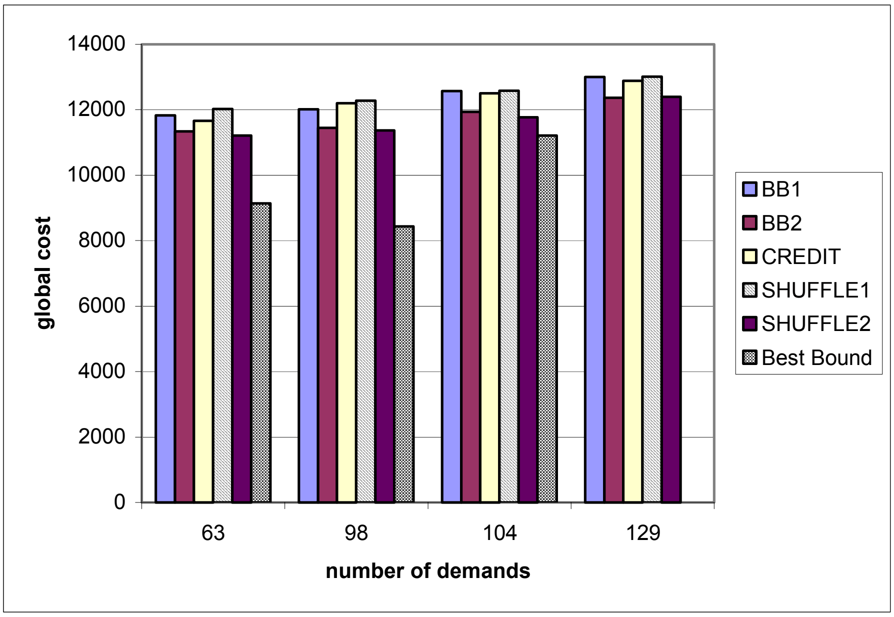
– 33 existing sites and 14 potential sites,
– 32 existing links and 39 potential links.

Demands are randomly generated.



The previous figure is a graphical representation of a solution. It shows the network state in different dates. The first image is (at right) the initial network with potential links in clear. The two others illustrate the evolution of the network during the time. We can notice the appearance of new sites and the increase of capacities of the links (the thickness of links is proportional to their capacity).

The following graph illustrates the comparison of the various exploration strategies applied to the presented search tree. Every search is executed with a five minutes time contract.



BB1 is our search scheme with a static heuristic. BB2 is the same search scheme with a dynamic heuristic that builds a path in the network. CREDIT, Shuffle1 and Shuffle2 were applied to BB1. Best bound is a lower bound of the solutions. It is computed with the linear relaxation of a linear program. It helps us to estimate the quality of our solutions.

We test the different algorithms on four instances of different sizes. The CREDIT partial search significantly improve the solutions. The Shuffle1 method never improves the first solution. The best methods take into account the problem structure (BB2 andShuffle2). The combination of both methods does not give better solutions.

6. Conclusion

This application provides a good benchmark for our framework. This framework dedicated to on-line optimization allows to overcome a weakness of the CP approach: the resolution in limited time, while keeping the declarative properties of the CP paradigm.

This framework makes it easier to try new combinations of search limits and new temporal strategies and speeds up the development process of customized partial search algorithms. The code is clearer and more concise when using the search primitives. The use of templates of search facilitates the re-use and the capitalization of the algorithms. It is a good prototyping tool for this kind of problem.

In this application, the partial search showed its interest in limited time. The use of large neighborhood adapted to the problem structure improved the solutions quality in a significant way. This is a very generic algorithm: it was already applied to another problem: missions of the Earth Observation Satellites management, the ROADEF challenge 2003 subject.

7. Bibliography

[Applegate&Cook 1991] D. Applegate, B. Cook. A computational study of the job shop scheduling problem. Operations Research Society of America, 3(2), 1991.

[Beldiceanu & al. 1998] Beldiceanu N., E. Bourreau, H. Simonis, and D. Rivreau (1998). Introduction de métaheuristiques dans CHIP. In Proc. of MIC-98.

[Bertsekas 1998] Dimitri P. Bertsekas, Network Optmization, 1998.

[de Givry&Jeannin 2003] de Givry S., Jeannin L. ToOLS : A library for partial and hybrid search methods, à paraître dans CP-AI-OR 2003.

[EOLE 2001] Eole consortium. Towards an on-line optimisation framework. CP-01 workshop On-Line combinatorial problem solving and Constraint Programming (OLCP'01), 2001.

[Hentenryck 1989] P. Van Hentenryck. Constraint Satisfaction in Logic Programming. The MIT Press, 1989.

[Hentenryck99] P. Van Hentenryck. OPL: The Optimization Programming Language. The MIT Press,Cambridge,Mass.,1999.

[Laburthe & al. 1998] Laburthe, F., P. Savéant, S. de Givry, and J. Jourdan (1998). Eclair: A library of constraints over finite domains. Technical Report technical report ATS 98-2, Thomson-CSF LCR, Orsay, France.

[Le Pape & al. 2002] Le Pape C., Perron L., Régin J., Shaw P, Robust and parallel solving of a design problem. In proc of CP 2002.

[Loudni&Boizumault 2001] Une approche hybride pour la résolution des VCSP en contexte anytime. In Proc. of JNPC-2001, Toulouse, 2001.

[Perron 2002] Perron L., Parallel and random solving of a network design problem. *In workshop on Random Search of AAAI conference*, 2002.