A LocalSolver model for the positioning of fuel assemblies

Jean-Yves Lucas[1], Didier Marcel[2],Thierry Benoist[3], Frédéric Gardi[3], Romain Megel[3]

[1] EDF R&D, 1. av. du Général de Gaulle, 92140 Clamart
jean-yves.lucas@edf.fr
[2] ENSTA ParisTech / CNAM, 828, Boulevard des Maréchaux, 91762 Palaiseau Cedex
[3] LocalSolver, 24 Avenue Hoche, 75008 Paris
{tbenoist,fgardi,rmegel}@localsolver.com

Mots-clés *: Quadratic Assignment, LocalSolver*

1 Introduction

The fuel in the heart of a nuclear reactor is made of many elementary units called "fuel assemblies". The number of these assemblies depends on the power rating of the reactor, about 150 for the first generations of PWRs to about 240 for the most recent (EPR). During a production cycle, each assembly is provided on its upper part with a special tool called cluster, of a particular type. There are four types of clusters. Each assembly shall be equipped with a cluster of the required type, depending on its position in the heart of the reactor. Nuclear fuel exhausting gradually over the production cycle, it is necessary to provide regularly (about once a year) a partial replacement of assemblies either by third by quarter during shutdowns for maintenance and refueling. For that all assemblies of the heart are discharged from the reactor and placed in the cooling pool. Third (or fourth) most worn assemblies will remain in the pool after the end of the shutdown, and will be replaced by new assemblies, already present in the pool before the unloading phase. To provide each assembly of the next production campaign with the required cluster, a robot implements an ordered sequence of permutations of clusters. Following this, each assembly (new or not) entering the heart for next production cycle has been equipped with a cluster of good type. Given the visiting order, it is necessary to position them in the pool in such a way as to minimize the total time of the trip. Indeed, time is precious during these stops: one day of downtime costs about one million euros what makes the optimization of this task (which can be on the critical path) very important. This study addresses this "positioning problem".

This assemblies positioning problem can be formalized as a quadratic assignment problem (QAP). In its classical form, it consists in assigning N objects to P postions ($P \geq N$), minimizing a cost function related to the relative positions of objects. This problem is known to be NP-complete. However our problem has a interesting specificity: a cost is assigned to the relative positioning of assemblies i and j only if these assemblies are visited consecutively in the trip of the robot.

2 Modeling and solving

This problem is not naturally modeled as a Mixed Integer Program. Some linearizations are available in the literature but they turned out to be ineffective on large scale real world instances as the ones that we are considering here. This finding lead us to use a local search approach, with LocalSolver. This mathematical programming solver is based on an *model & run* paradigm that is

to say that it takes as input a model in a simple mathematical formalism and the resolution is then completely automatic. For example, in this case, the variables and constraints are written in a few lines.

```
function model() {
    //variables 0-1: z[a][p] = 1 if assembly a is assigned to position p
    z[a in 1..nb_assemblies][p in 1..nb_positions] <- bool();

    //assignment: one position per assembly and at most one assembly per postion
    for[a in 1..nb_assemblies]  constraint sum[p in 1..nb_positions] (z[a][p]) == 1;
    for[p in 1..nb_positions]  constraint sum[a in 1..nb_assemblies](z[a][p]) <= 1 ;

    // coordinates (x[a],y[a]) of the position assigned to assembly a,
    // given the coordinates (X[p],Y[p]) of positions (data)
    x[a in 1..nb_assemblies] <- sum[p in 1..nb_positions](X[p]* z[a][p]);
    y[a in 1..nb_assemblies] <- sum[p in 1..nb_positions](Y[p]* z[a][p]);

    // duration of the trip from  S[i-1] to S[i] where S is the sequence of assemblies
    // that is to say the ordering assigned to the robot (data).
    time[i in 1..n] <- max(abs(x[S[i])-x[S[i-1]]), abs(y[S[i])-y[S[i-1]]));

    // objective function
    minimize sum[i in 1..n](time[i]);
}
```

One of the strengths of LocalSolver is the ability for the user to use any numeric expressions (possibly via user-defined functions). In our case for example, the operators *max* and *abs* allowed us to define the Chebyshev distance in a very natural way. This is obviously more complicated in linear programming and requires for example the addition of many binary variables.

On real world instances, the number of clusters to swap went from 157 to 241, the number of assemblies (including new assemblies) from 209 to 321, the number of elementary movements of the robot was generally between 200 and 300. The size of the problem is huge, even if finding a feasible solution is relatively easy since any assignment is valid. Nevertheless finding a (very) good solution is very difficult. On different data sets, we experimented running times ranging from a few seconds to 1 hour. For each instance, solutions have been found in the first seconds of the research, and then, they have been steadily improved until the time limit of one hour that we had set. The improvement from the first solutions to the best solution obtained after one hour ranges from 3% to 7%. On real-world instances of a complex combinatorial problem, LocalSolver has demonstrated its ability to provide solutions in a time compatible with the constraints of the business process.

LocalSolver follows the claims made by its designers. It was able to adapt and to provide good-quality solutions to the problem of placement of nuclear fuel assemblies in pools, in reasonable running times, short or even very short, on a standard computer. His strength relies on the number of iterations and moves performed. It was able to compete with a simulated annealing algorithm which, however, took into account the structure of the problem. It is a serious way to resolve the quadratic assignment problems encountered by EDF.