An ASP-based System for Team-building in the Gioia-Tauro Seaport

G. Grasso[1], S. Iiritano[2], N. Leone[1], V. Lio[2], F. Ricca[1], and F. Scalise[3]

[1] Dipartimento di Matematica, Università della Calabria, 87030 Rende, Italy
{leone,ricca}@mat.unical.it
[2] Exeura Srl, Via Pedro Alvares Cabrai - C.da Lecco 87036 Rende (CS), Italy
{salvatore.iiritano,vincenzino.lio}@exeura.com
[3] ICO BLG Logistics Automobile Italia SPA - Gioia Tauro, Italy

Abstract. We have developed a system based on Answer Set Programming (ASP) for the automatic generation of the teams of employees in the seaport of Gioia Tauro. The problem here is to generate a correct allocation of the available personnel of the international seaport of Gioia Tauro in such a way that the right processing of the shoring cargo boats is guaranteed. To this end several constraints have to be satisfied. Depending on the size and the load of cargo boats, an appropriate number of employees of different skills is required. The selection of the employees and the role they play in the team (each employee might cover several roles according with his/her skills) are subject to many conditions (e.g., fair distribution of the working load, tournament of the heavy/dangerous roles, etc.) The system can build new teams, complete the allocation automatically when some key employees are fixed manually, and check the correctness of manually generated team, providing proper explainations if no correct team can be generated. In this application, the domain is modeled by exploiting ASP and implemented by using the ASP system DLV. A set of suitably defined logic programs is exploited for finding the desired allocation. The pure declarative nature of the language allowed us for refining and tuning both problem specifications and encodings together while interacting with the stakeholders of the seaport. It is worth noting that the possibility of modifying (by editing text files) in a few minutes a complex reasoning task (e.g. by adding new constraints), and testing it "on-site" together with the customer was a great advantage of our approach. The system is currenty exploited by the ICO BLG company at the seaport of Gioia Tauro.

1 Scenario

The seaport of Gioia Tauro (http://www.portodigioiatauro.it) is the largest transshipment terminal of the Mediterranean Sea. Historically, container transshipments are the main activity of the seaport (related problems were subject of extensive research [4]); recently, Gioia Tauro has become also an automobile hub. Automobile logistics is carried out by the company ICO B.L.G. (a subsidiary of the B.L.G. Logistics Group - http://www.blg.de). Several ships of different size shore the port every day, transported vehicles are handled, warehoused, if necessary technically processed and then delivered to their final destination. The goal is to serve them as soon as possible. Data regarding the shoring boats (arrival/departure date, number and kind of vehicles, etc.), is available

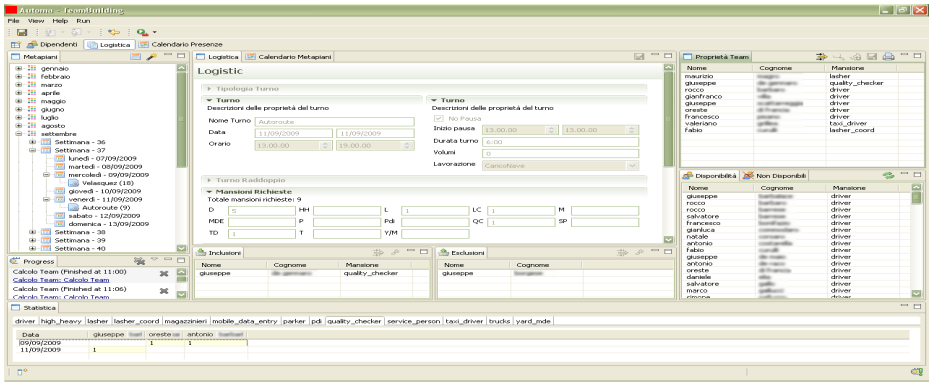

Fig. 1. The Team-builder Graphic User Interface

at least one day in advance; and, suitable teams of employees have to be arranged for the purpose. Teams are subject to many conditions. Some constraints are imposed by the contract (e.g. an employee cannot work more than 36 hours per week, etc.), some other by the required skills. Importantly, heavy/dangerous roles have to be turned over, and a fair distribution of the workload has to be guaranteed. Once the information regarding shoring boats is received, the management easily produces a meta-plan specifying the number of employees required for each skill; but a more difficult task is to assign the available employees to shifts and roles (each employee might cover several roles according with his/her skills) in such a way that the above-mentioned constrains can be satisfied every day. The impossibility of allocating teams to incoming boats might cause delays and/or violations of the contract with shipping companies, with consequent pecuniary sanctions for B.L.G. Thus, team building is a crucial management task.

2 ASP-based team builder

Answer Set Programming (ASP) [1] is a purely-declarative logic programming language allowing for disjunction and nonmonotonic negation. We exploited ASP for developing a team builder and, in this Section, we give a flavor of its working principles. A simplified version of the kernel part of the employed ASP program is reported below:

$$(r)\ assign(Em, Sh, Sk) \vee nAssign(Em, Sh, Sk) :- skill(Em, Sk), metaPlan(Sh, Sk, \_, D), not\ absent(Em), not\ manuallyExcluded(Em), workedHours(Em, Wh), Wh + D \leq 36.$$

$$(c_1) :- metaPlan(Sh, Sk, EmpNum, \_), \#\mathrm{count}\{Em : assign(Em, Sh, Sk)\} \neq EmpNum.$$

$$(c_2) :- assign(Em, Sh, Sk1), assign(Em, Sh, Sk2), Sk1 \neq Sk2.$$

$$(c_3) :- wstats(Em1, Sk, \_, LastTime1), wstats(Em2, Sk, \_, LastTime2), LastTime1 > LastTime2, assign(Em1, Sh, Sk), not\ assign(Em2, Sh, Sk).$$

$$(c_4) :- workedHours(Em1, Wh1), workedHours(Em2, Wh2), threshold(Tr), Wh1 + Tr < Wh2, assign(Em1, Sh, Sk), not\ assign(Em2, Sh, Sk).$$

$$(r_{aux})\ workedHours(Em, Wh) :- skill(Em, \_), \#\mathrm{count}\{H, Em : wstats(Em, \_, H, \_)\} = Wh.$$

The inputs are: the employees and their skills (predicate *skill(employee, skill)*; a meta-plan specification (predicate *metaPlan(shift, skill, neededEmployees, duration)*); weekly statistics specifying for each employee both the number of worked hours per skill and the last allocation date (predicate *wstat(employee, skill, hours, lastTime)*); absent employees (predicate *absent(employee)*; and employees excluded by a management decision (predicate *manuallyExcluded(employee)*). Following the guess&check

programming methodology [2], the disjunctive rule r generates the search space by guessing the assignment of a number of available employees to the shift in the appropriate roles. Absent or manually excluded employees, together with employees exceeding the maximum number of weekly working hours are automatically discarded. Then, admissible solutions are selected by means of constraints: c_1 discards assignments with an wrong number of employees in some skill; c_2 avoids that an employee covers two roles in the same shift; c_3 implement the tournament of roles; and c_4 guarantees a fair distribution of the workload. r_{aux} computes the total number of worked hours per employee. (If no plan can be generated, then the system suggests the user to relax some constraints). Note that, only the kernel part of the employed logic program is reported here (in a simplified form), and many other constraints were developed, tuned and tested.

3 The system

The team-building system integrates the ASP system DLV [2] and features a Graphical User Interface (GUI) developed in Java. In particular, the GUI is based on the Rich Client Platform (RCP) technology; whereas, reasoning services and and data-storage features are implemented with OntoDLV [3], an ontology management and reasoning system based on DLV. The GUI combines in a single frame all the controls (see Figure 1). A tree-shaped calendar (displayed on the left) allows for browsing and scheduling working activities. Meta-plans specifications, usually identified by the name of the corresponding cargo boats (e.g. Velasquez, Autoroute), are the leafs of the tree, which can be added or removed by right-clicking on their name and selecting the proper command from a context-menu. Meta-plans information (ship arrival and departure date, available processing time and requested skills) is displayed in (and is modified by editing) the “Logistics” panel. Below, the “Inclusion” and “Exclusion” panels allows for pre-assinging (or excluding) specific employees from the team. To run the system the user selects a meta-plan (from the tree), right-clicks on it (a context-menu appears) and chooses the “run” item. Input information and personnel statistics are fed into the DLV system and the result is displayed on the top-right panel (“Team Properties”). The computed team can be also modified manually, and the system is able to verify if the manually-modified team still satisfies the constraints. In case of errors, causes are outlined and suggestion for fixing a problems proposed. The interface gives full control on the status o the seaport-staff: available/unavailable personnel is listed on the bottom-right panel, and the allocation statistics are reported in the bottom panel.

References

1. Gelfond, M., Lifschitz, V.: Classical Negation in Logic Programs and Disjunctive Databases. New Generation Computing **9** (1991) 365–385
2. Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., Scarcello, F.: The DLV System for Knowledge Representation and Reasoning. ACM TOCL **7**(3) (2006) 499–562
3. Ricca, F., Gallucci, L., Schindlauer, R., Dell’Armi, T., Grasso, G., Leone, N.: OntoDLV: an ASP-based system for enterprise ontologies. Journal of Logic and Computation (2009)
4. Vacca, I., Bierlaire, M., and Salani, M.: Optimization at Container Terminals: Status, Trends and Perspectives. In Proc. of Swiss Transport Research Conference, September 12-14, 2007