A Constraint-based Model for Multi-objective Repair Planning

I. Barba, C. Del Valle and D. Borrego
Dpto. Lenguajes y Sistemas Informáticos, Universidad de Sevilla
{irenebr,carmelo,dianabn}@us.es

Abstract

This work presents a constraint based model for the planning and scheduling of disconnection and connection tasks when repairing faulty components in a system. Since multi-mode operations are considered, the problem involves the ordering and the selection of the tasks and modes from a set of alternatives, using the shared resources efficiently. Additionally, delays due to change of configurations and transportation are considered. The goal is the minimization of two objective functions: makespan and cost. The set of all feasible plans are represented by an extended And/Or graph, that embodies all of the constraints of the problem, allowing non reversible and parallel plans. A simple branch-and-bound algorithm has been used for testing the model with different combinations of the functions to minimize using the weighted-sum approach.

1. Introduction

Constraint-based techniques have been used successfully to solve a wide scope of applications related to scheduling problems. Some of the extensions to scheduling, such as alternative resources and process alternatives, lead to models that are closer to planning [13]. Also, the AI planning community has done several efforts to extend classical planning techniques to treat resources and time constraints. There is an increasing interest for integrating planning and scheduling since real-world problems involve both of them [1]. Some of the applications involving such issues are maintenance and repair planning, where there may be a cascading set of choices for actions, facilities, tools or personnel, which affect different features of the plan, such as duration or cost [13]. Assembly and disassembly planning involve the identification, selection and sequencing of operations, which can be specified by their effects on the components. In other context, disassembly planning has been object of different studies, such as maintenance or repair purposes [11]. Different techniques have been used for solving those problems, from mathematical programming to a variety of methods related to artificial intelligence [10].

Many problems, such as decision, planning and scheduling problems, can involve multiple conflicting objectives that should be considered at the same time. In multi-objective optimization problems, that have been studied for several decades [2], usually no unique solution exists but a set of nondominated solutions can be founded. These solutions are also known as Pareto optimal solutions (to obtain a better feasible solution in one of the objectives, it is necessary to deteriorate, at least, another one objective). Two of the typical objectives pursued in planning and scheduling problems, used in this work, are the minimization of the total time and cost of the resulting plan. Sometimes, these two objectives are in conflict, so it is not easy to find a feasible plan that is good in both of them. Related works can be found, such as [8], that proposes a hybrid algorithm for finding a set of nondominated solutions of multi-objective flowshop scheduling problems.

This work presents a CSP (Constraint Satisfaction Problem) model for solving a planning problem corresponding to the optimal sequencing of disconnection and connection tasks for repair or substituting faulty components. The objective is the minimization of the total reparation time and cost when executing the plan in a generic multiple machine environment, considering different factors that can have an influence on it: durations and cost of tasks, shared resources and estimations of time and cost needed for doing auxiliary operations, such as the transportation of intermediate subsystems between different machines, and set-up operations such as changes of configurations in machines. In order to work with a more flexible environment, it is considered that the tasks can be executed in several operating modes (multi-mode project scheduling [9]), each one using a different machine or configuration, and possibly different duration and cost.

The rest of the paper is organized as follows: Section 2 details the considered repair problem, Section 3 summarizes the main aspects of Constraint Programming, Section 4 states the CSP model for planning the substitution or reparation of faulty components, Section 5 describes the main approaches to solving multi-objective optimization problems, Section 6 shows some experimental results and, finally, Section 7 presents some conclusions and future work to be developed starting from the model proposed.

2. The Repair Planning Problem

In order to repair a (previously detected) faulty component, a sequence of disconnection tasks must be executed to get it. After that, a repair action would substitute or repair the component, and then a set of connection tasks must reconnect the system.

2.1. The And/Or Graph

There are some representations for the repair planning problem. One of the most used is through And/Or graphs [6], that allows to represent the set of all feasible connection and disconnection plans in a natural way. In this representation two kinds of nodes can be distinguished:

- Or nodes: correspond to subsystems, the top node corresponds to the complete system, and the leaf nodes correspond to the individual components.
- And nodes: correspond to the connection tasks joining the subsystems of its two Or nodes below it producing the subsystem corresponding to the Or node above it; and the disconnection tasks, that decomposes the subsystem above it to obtain the two subsystems below it.

For the same Or node, there can be several And nodes (tasks) below it, representing different alternatives to connect/disconnect the corresponding subsystem.

In these graphs, each connection/disconnection plan is associated to a tree, that is an And/Or path starting at the root node and ending at the leaf nodes. An important advantage of this representation, used in this work, is that the And/Or graph shows the tasks that can be executed in parallel (see Figure 1). Furthermore, both precedence constraints and those related to the selection of tasks for obtaining a correct plan, can be easily obtained from this representation. As explained in Section 4, an extension of this representation will allow a direct mapping from the planning problem to a CSP, in order to be solved using Constraint Programming (Section 3).

2.2. Types of Tasks in a Repair Plan

A feasible repair plan can be seen as a set of tasks that have to be executed, starting with the disconnection of the complete system, and finishing with the connection of the repaired system. There are two kinds of tasks:

- Connection/Disconnection tasks: are executed on an established machine with a particular configuration to obtain one (connection) or two (disconnection) subsystems.
- Auxiliary tasks: due to the use of shared resources and different machines; it is considered two kinds of operations: *set-up operations*, that change the configuration of a machine when two successive tasks with different configuration use that machine; and *transportation operations*, that transport the subsystems between machines when the machine where the subsystem is obtained is different from the machine where is required.

In this work, it is considered a duration and cost associated with the execution of each task, as it is explained in Section 4.

2.3. Multi-mode Tasks

As stated before, it is considered that the tasks can be executed in more than one operating mode, each one using a different machine or configuration and possibly different duration and cost. Taking into account this, there can be several options to connect two subsystems to obtain another one, or disconnect one subsystem to obtain two ones.

In this work, it is considered that each task with an operating mode, corresponds to a different And node in the graph (example T'_2 and T'_3 in Figure 1).

2.4. Some Considerations

Usually, some different properties are fulfilled, and considering them can simplify solving the problem. Some of them are taken into account in this work as gathering in the following definitions. First, a repair graph is a sub-graph of the And/Or graph which only contains the connection and disconnection tasks (and the corresponding subsystems) that could be necessary to repair some components, according to the simplified model considered. Another important consideration is that a connection (disconnection) task T is reversible if its corresponding disconnection (connection) task T' is feasible. Lastly, a reversible plan is a tree of the repair graph that only contains reversible tasks, so that for each disconnection task, its reverse connection task is included.

The planning model developed in the current work supposes two assumptions: (A1) All tasks are reversible and (A2) Subsystems that do not include the faulty components are not disconnected. Taking into account (A1) and (A2), in the connection process, other subsystems different from the ones generated by the disconnection process can appear, depending on how they are joined. Moreover, disconnection tasks only handle subsystems that contain the faulty component, whereas connection tasks handle subsystems that may contain or not the faulty component.

In general, plans are not linear sequences of tasks, unlike reversible plans. Although the disconnection process is linear, the connection can contain tasks that may execute in parallel with others. Moreover, it is possible that the connection process starts before the disconnection process has finished and there may be a parallel execution of the two types of tasks.

3. Constraint Programming

Constraint Programming (CP) has been evolved in the last decade to a mature field because, among others, of the

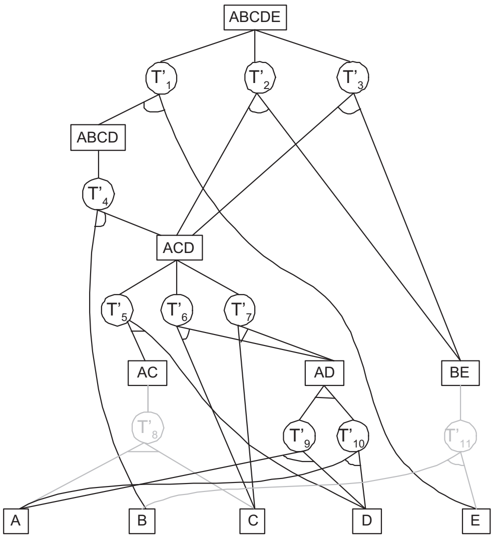

Figure 1. The simplified And/Or graph

use of different generic and interchangeable procedures for inference and search, which can be used for solving different types of problems [12]. A Constraint Satisfaction Problem (CSP) is defined by a set of variables, the set of domains of values for each variable and a set of constraints. Each constraint involves some variables and specifies the allowable combinations of values for them. A solution is defined by an assignment of values to all the variables, being feasible if it does not violate any constraint. Constraint Optimization Problems (COPs) require a solution that optimize an objective function.

There exists a wide scope of mechanisms used to solve CSPs and COPs, that can be classified as search or consistency algorithms [12]. Search algorithms are based on the exploration of the solution space to find a solution or to prove that there no exist any one. It is possible to differentiate between systematic algorithms and local search algorithms: systematic algorithms generally explore a search tree which is based on the possible values for each of the variables of the CSP problem. On the other hand, local search algorithms, in general, perform an incomplete exploration of the search space by repairing infeasible complete assignments or trying to improve the objective value. On the other hand, Consistency algorithms consist on removing inconsistent values from the variables domain. One way to accomplish this is evolving from the initial problem towards equivalent problems whose solution space is smaller, so it is easier to solve.

Once a problem is modelled by a CSP, a generic or specialized CSP solver can be used in order to obtain the required solution.

4. The CSP Model

According to the problem stated in the previous section, the time and resource constraints, typical from scheduling, would be modified to conditional constraints taking into account that tasks (and subsystems) may not appear in the solution. Most of the ideas are taken from [4], but the assumptions considered in this work will result in modifying most constraints and in adding others: apart from the optimization of the duration, the minimization of the total cost is pursued, resulting in a multi-objective optimization problem, and multi-mode tasks are considered.

Taking into account the assumptions (A1) and (A2), the And/Or graph can be simplified (see Figure 1), removing those And nodes below the Or nodes corresponding to subsystems which do not contain the faulty component. Figure 2 shows an example of this representation.

4.1. Variables of the CSP

Four kinds of CSP variables have been defined: selection, resource, time and cost variables.

Selection variables. For each And node, two boolean variables represent if the connection and disconnection tasks are selected for the solution, $s(T)$ and $s(T')$ respectively. Furthermore, for each Or node, two boolean variables represent if the subsystem S appears in the connection and disconnection processes, $s(S)$ and $s'(S)$ respectively.

Resource variables. For each And node, $M(T)$ and $M(T')$ show the machines used, and $Cf(T)$ and $Cf(T')$ are the necessary configuration on them for the connection and disconnection tasks respectively. These values are data of the problem. On the other hand, the machine where a subsystem is obtained after the corresponding disconnection and connection task, are represented by the variables $m'(S)$ and $m(S)$ respectively, that are variables of the CSP.

Time variables. For each And node, the durations of the associated tasks $Dur(T)$ and $Dur(T')$ are established. Due to the auxiliary operations, $\Delta_{cht}(M, Cf, Cf')$ denotes the time needed for changing the configuration of the machine M from Cf to Cf', and $\Delta_{mov}(S, M, M')$ denotes the time needed for transporting the subsystem S from machine M to machine M'. Finally, a component C to be repaired is associated to a temporal delay $\Delta_{subst}(C)$, corresponding to the reparation or substitution of the faulty component. These values are data of the problem.

On the other hand, for each And node, the CSP variables related to the time are: its starting times, $t_i(T)$ and $t_i(T')$ and ending times, $t_f(T)$ and $t_f(T')$. For each Or node, the times when it is obtained after connection, $t_{OR}(S)$, and disconnection, $t'_{OR}(S)$.

Cost variables For each And node, it is considered: its connection and disconnection cost, $Cost(T_i)$ and $Cost(T'_i)$ respectively. Regarding to the auxiliary operations, $Cost_{cht}(M, Cf, Cf')$ denotes the cost of changing the configuration of the machine M from Cf to Cf', and $Cost_{mov}(S, M, M')$ denotes the cost of transporting the subsystem S from machine M to machine M'. Furthermore, a component C to be repaired is associated to a cost $Cost_{subst}(C)$, corresponding to the reparation or substitution of the faulty component.

On the other hand, for each And node, the selection of the corresponding task T may be associated some additional costs, as explained in Section 4.3 : first, the variable $cost_{mov}(T_i)$ represents the possible costs associated to the movement of subsystems to $m(T)$, if this machine is different of the one where the subsystem were previously obtained; and secondly, the variable $cost_{cht}(T_i)$ represents the possible costs of change of configuration, if $m(T)$ has been previously used with a different one. These variables are linked to the And nodes because the costs are due to the selection of the corresponding task.

Finally, a variable that represents the total cost of a plan, $cost_{total}$, has been used in order to minimize this objective function.

4.2. The extended And/Or graph

The original And/Or graph has been extended, so that the new representation includes all the constraints involved in the problem, adding new types of links between And nodes. The new links represent non-precedence constraints: due to the use of shared resources by the tasks (relations of type 6 below) and due to the change of configurations in the machines (relations of type 5 below). Figure 2 shows the extended and simplified repair And/Or graph resulting for a system consisting in 5 components, $ABCDE$, when substituting component D. Although all the leaf nodes generated in the disconnection process must be present in the connection part of the solution, the same is not true for the intermediate subsystems, which will appear or not depending on the connection tasks selected. A typical objective for such a problem would be the minimization of the elapsed time of the plan, that is the time when the system is reconnected after the reparation, given by the variable $t_{OR}(ABCDE)$ for the example used. Another important issue is the total cost of the complete repair plan, given by the variable $cost_{total}$, previously defined. In this work, a multi-objective optimization is pursued, encompassing both of them.

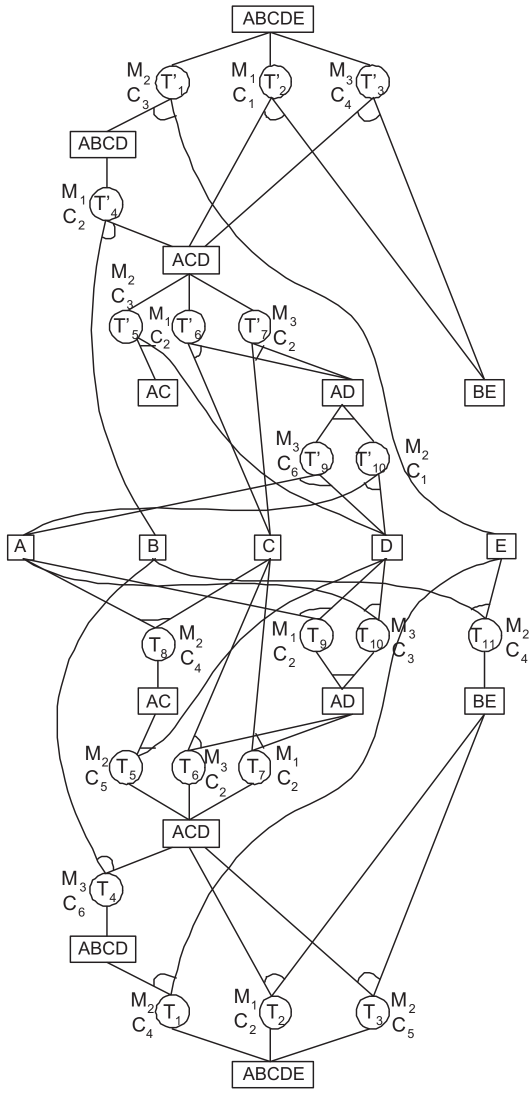

Figure 2. The simplified repair And/Or graph

4.3. Types of constraints

In this work, 6 types of relations are considered (see Figure 3), each one representing a link or component of the extended And/Or graph (see Figure 4):

Relations of type (1) collect the relation between the information from an Or node and the And nodes below it in the original And/Or graph.

Relations of type (2) consider the durations of connection and disconnection tasks, and correspond to the relationships between the starting and ending times of the connection and disconnection tasks.

Relations of type (3) collect the relation between the information from an And node and the (two) Or nodes below it in the original And/Or graph.

Relations of type (4) consider the relation between the selection of an Or node and all the And nodes above it (possibly only one) in the original And/Or graph.

Relations of type (5) are due to the delay needed for a change of configuration in a machine between the executions of two successive tasks using the same machine with precedence constraints among them. Those include the relations between reverse disconnection and connection tasks. Notice that, for a particular repair plan, it is only needed relating each task to its closest successor one that uses the same machine in the And/Or tree.

Relations of type (6) consider the relation between some tasks that use the same resource.

Types (1), (2), (3) and (4) come from the relations between the nodes included in the original graph, while types (5) and (6) come from the use of (same or different) resources by the different tasks, and they are related to new links between tasks in the extended And/Or graph.

Taking into account the variables of the proposed CSP model (see Section 4.1), a classification about the types of constraints can be done: selection, resource, time and cost constraints. In a previous work [4], the first three kinds

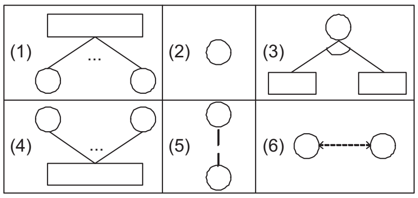

Figure 3. Types of Relations

of constraints are taking into account, so only an abstract about them is shown. However, cost constraints have been considered for the first time in this work, so an extended explanation can be seen.

Selection Constraints collect the relations between the boolean variables that represent if the tasks are selected for the solution and the subsystems appears in the repair process. A special case is for the complete system and for the faulty component, which always will be part of the solution, so $s'(ABCDE) = s(ABCDE) = s'(D) = s(D) = true$.

Related to relations of type (1), the next constraints include the selection of disconnection tasks T' and connection tasks T with that of subsystems, expressed through the XOR operator: $s'(S) \Leftrightarrow XOR_{T'_i \in succ(S)}(s(T'_i))$ and $s(S) \Leftrightarrow XOR_{T_i \in succ(S)}(s(T_i))$ (example $s'(ABCDE) \Leftrightarrow (XOR(s(T'_1), s(T'_2), s(T'_3)))$.

Related to relations of type (3), the obligatory selection of the two Or nodes if the And node is selected: $s(T') \Rightarrow s'(S_1) \wedge s'(S_2)$ and $s(T) \Rightarrow s(S_1) \wedge s(S_2)$ (example $s(T'_1) \Rightarrow s'(ABCD) \wedge s'(E)$).

Related to relations of type (4), the next constraints include the selection of disconnection tasks T' and connection tasks T with that of subsystems, expressed through the XOR operator: $s'(S) \Leftrightarrow XOR_{T'_i \in pred(S)}(s(T'_i))$ and $s(S) \Leftrightarrow XOR_{T_i \in pred(S)}(s(T_i))$ (example $s(ACD) \Leftrightarrow XOR(s(T_2), s(T_3), s(T_4))$).

Resource Constraints consider the relations between the machines used in the connection and disconnection tasks, and the machines where the subsystems are obtained after them.

Related to relations of type (1), the machine m where a subsystem is generated after a connection task is the machine used by this task: $s(T_i) \Rightarrow m(S) = M(T_i)$ (example $s(T_{10}) \Rightarrow m(AD) = M(T_{10})$).

Related to relations of type (3), the machine m' where a subsystem is generated after a disconnection task is the machine used by this task: $s(T'_i) \Rightarrow m'(S_1) = m'(S_2) = M(T'_i)$ (example $s(T'_9) \Rightarrow m'(A) = m'(D) = M(T'_9)$).

Time Constraints collect the relations between the start and the end times of the tasks, and the time when the subsystems are obtained. Initially, $t'_{OR}(ABCDE) = 0$.

Related to relations of type (1), these constraints establish the disconnection times t'_{OR} and connection times t_{OR} of Or nodes related to the start times of the disconnection tasks or the end times of the connection tasks: $s(T'_i) \Rightarrow t_i(T'_i) \geq t'_{OR}(S) + \Delta_{mov}(S, m'(S), M(T'_i))$ and $s(T_i) \Rightarrow t_f(T_i) = t_{OR}(S)$

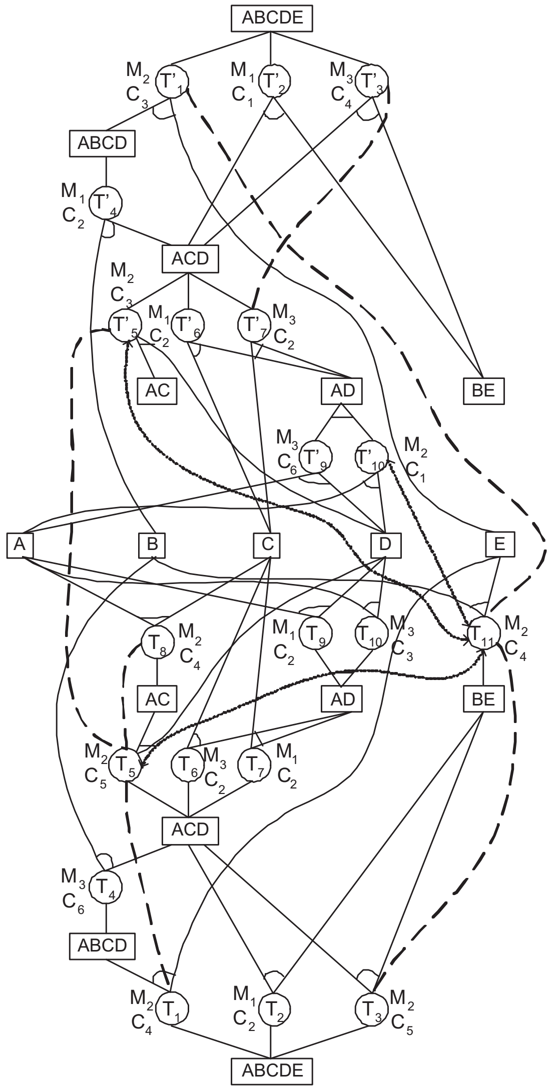

Figure 4. The simplified repair And/Or graph with relations (5) and (6) between tasks

(example $s(T'_1) \Rightarrow t_i(T'_1) \geq t'_{OR}(ABCDE) + \Delta_{mov}(ABCDE, m'(ABCDE), M(T'_1))$).

Related to relations of type (2), these constraints consider the end time of the tasks related to the start time and the durations of them: $s(T'_i) \Rightarrow t_f(T'_i) = t_i(T'_i) + Dur(T'_i)$ and $s(T_i) \Rightarrow t_f(T_i) = t_i(T_i) + Dur(T_i)$ (example $s(T'_1) \Rightarrow t_f(T'_1) = t_i(T'_1) + Dur(T'_1)$).

Related to relations (3), the next constraints include the equality constraint between the disconnection times of the Or nodes t'_{OR} and the end time of a disconnection task T' above them in the original And/Or graph: $s(T'_i) \Rightarrow t_f(T'_i) = t'_{OR}(S_1) = t'_{OR}(S_2)$, and the precedence between the connection time of the Or nodes t_{OR} and the start times of connection task T (And nodes), and considering the possible delays due to the transportation of subsystems if the two successive tasks involving it use different machines: $s(T_i) \Rightarrow$

$t_i(T_i) \geq t_{OR}(S_1) + \Delta_{mov}(S_1, m(S_1), M(T_i))$ and $s(T_i) \Rightarrow t_i(T_i) \geq t_{OR}(S_2) + \Delta_{mov}(S_2, m(S_2), M(T_i))$ (example $s(T'_{10}) \Rightarrow t_f(T'_{10}) = t'_{OR}(A) = t'_{OR}(D)$).

Related to relations of type (5), these constraints establish that for a task T_i, and its closest predecessor task T_j using the same machine m, taking into account the possible change of configuration: $(s(T_i) \wedge s(T_j)) \Rightarrow t_i(T_j) \geq t_f(T_i) + \Delta_{cht}(m, Cf(T_i), Cf(T_j))$ (example $(s(T'_1) \wedge s(T'_{10})) \Rightarrow t_i(T'_{10}) \geq t_f(T'_1) + \Delta_{cht}(M2, Cf(T'_1), Cf(T'_{10})))$.

Moreover, since the solution may contain non-reverse tasks, each disconnection task must be related to each closest successor connection task that uses the same machine. Furthermore, when both tasks use the same configuration, the resulting constraint is superfluous and can be eliminated.

For each two tasks T_i and T_j requiring the same machine m, with no precedence constraint among them, and which may belong to the same repair plan, the constraints of type (6) express the two possible orders of execution of the tasks: $(s(T_i) \wedge s(T_j)) \Rightarrow (t_i(T_i) \geq t_f(T_j) + \Delta_{cht}(m, Cf(T_j), Cf(T_i)) \vee t_i(T_j) \geq t_f(T_i) + \Delta_{cht}(m, Cf(T_i), Cf(T_j)))$ (example $(s(T_8) \wedge s(T_{11})) \Rightarrow t_i(T_8) \geq t_f(T_{11}) + \Delta_{cht}(M2, Cf(T_{11}), Cf(T_8)) \vee t_i(T_{11}) \geq t_f(T_8) + \Delta_{cht}(M2, Cf(T_8), Cf(T_{11})))$.

For the Or leaf nodes (including those that do not include the faulty component) t'_{OR} and t_{OR} are equals, except for the faulty component, in which the delay corresponding to the reparation is considered.

Cost Constraints: In the repair process of a component in a complete system, the cost of a plan can be established by the aggregated costs associated to the execution of the selected tasks. The total cost of selecting a task T_i involves:

- the execution cost of the task, $Cost(T_i)$ (related to relation (2))
- the cost associated to the possible movement of one or two subsystems from one machine to another, $cost_{mov}(T_i)$:
  - in disconnection tasks T'_i, it is necessary to take into account the possible movement of the subsystem related to the Or nodes above it in the original And/Or graph, related to relation (1), $cost_{mov}(T'_i) = Cost_{mov}(S, m'(S), M(T'_i))$
  - in connection tasks T_i, it is necessary to take into account the possible movement of the two subsystems related to Or nodes below it in the original And/Or graph, related to relation (3), $cost_{mov}(T_i) = Cost_{mov}(S_1, m(S_1), M(T_i)) + Cost_{mov}(S_2, m(S_2), M(T_i))$
- the possible cost associated to a change of configuration on $M(T_i)$, $cost_{cht}(T_i)$. If $M(T_i)$ has been used before by another task with a different configuration, it is necessary to change it. An additional complexity of the considered problem is that the cost of the change of configuration depends of the sequence of tasks for each machine, so there must be considered the **precedent task** executed on $m(T_i)$, being necessary to analyze two groups of tasks: set of possible immediate predecessors of T_i using the same machine (precedent tasks with Relation (5)); and set of tasks linked to T_i by the relation (6), explained in Section 4.3.

  Taking into account this, $cost_{cht}(T_i) = Cost_{cht}(M(T_i), Cf(PM(T_i)), Cf(T_i))$, where $PM(T_i)$ is the precedent task executed on $m(T_i)$. If T_i is executed the first on its machine, $cost_{cht}(T_i) = 0$.

On the other hand, the total cost of a plan can be defined as $cost_{total} = \sum_{T_i} s(T_i)(Cost(T_i) + cost_{mov}(T_i) + cost_{cht}(T_i))$.

In Table 1, some cost constraints of the And/Or graph of the Figure 4 are shown.

Table 1. Cost Constraints

Type	Constraint
(1)	$s(T'_1) \Rightarrow cost_{mov}(T'_1) = Cost_{mov}(ABCDE, m'(ABCDE), M(T'_1))$
(1)	...
(1)	$s(T'_{10}) \Rightarrow cost_{mov}(T'_{10}) = Cost_{mov}(AD, m'(AD), M(T'_{10}))$
(3)	$s(T_1) \Rightarrow cost_{mov}(T_1) = Cost_{mov}(ABCD, m(ABCD), M(T_1)) + Cost_{mov}(E, m(E), M(T_1))$
(3)	...
(3)	$s(T_{11}) \Rightarrow cost_{mov}(T_{11}) = Cost_{mov}(B, m(B), M(T_{11})) + Cost_{mov}(E, m(E), M(T_{11}))$
(5),(6)	$s(T'_{10}) \Rightarrow cost_{cht}(T'_{10}) = Cost_{cht}(M(T'_{10}), Cf(argmax_{Ta \in \{T'_1, T_{11}\}}\{T_f(T_a) \mid s(T_a) \wedge t_f(T_a) \leq t_i(T'_{10})\}), Cf(T'_{10}))$
(5),(6)	...
(5),(6)	$s(T_8) \Rightarrow cost_{cht}(T_8) = Cost_{cht}(M(T_8), Cf(argmax_{Ta \in \{T'_{10}, T_{11}\}}\{T_f(T_a) \mid s(T_a) \wedge t_f(T_a) \leq t_i(T_8)\}), Cf(T_8))$

Notice that the combinatorial character of the problem is due to the XOR constraints of types (1) and (4) and the disjunctive constraints of type (6). These types of constraints correspond, respectively, to the selection of alternative tasks and to the use of shared resources by them that are not related through precedence constraints.

5. Multi-objective Optimization

In order to solve multi-objective optimization problems, there are, basically, three approaches. One of them, used in this work, consists on defining a new objective function that can be optimized with single objective solvers, such as the weighted-sum method [2], that minimizes $\sum w_i f_i$, where $w_i \geq 0$ for each objective function

Table 2. Number of And, Or nodes (average)

	And/Or graph		Simplified And/Or graph			
Prob	#Or	#And	#Or	#And	#And'	#RP
30-1	348	630	223	327	240	1213
30-2	404	828	303	520	365	9200
30-3	415	863	310	546	384	12846
40-1	649	1518	433	833	575	23005
40-2	770	2143	621	1489	984	248408
40-3	756	2060	598	1400	925	197551

f_i considered. It is advisable to normalize the objectives with some scaling so that different magnitudes do not confuse the method. For the proposed model, the objective function can be $w_t t_{OR}(P) + w_c cost_{total}$. This method is easy to use and, if all the weights are positive, the minimum of the previous proposed objective function is always Pareto optimal. On the other hand, by varying the weights it is possible to analyze some characteristics of the objective functions, such as the grade of significance or discrimination. Another approaches can be used, such as optimize one of the objective functions constraining the other ones (i.e. ε-Constraint Method [5]) or work with a set of Pareto optimal solutions (i.e. Evolutionary Multi-objective Optimization [3]).

Once the variables and the constraints of the problem are defined, any of these methods can be used to solve the problem. According to the constraint programming paradigm, as instantiating the constrained variables, the state of the constraint propagation can be used in order to guide the search, apart from reducing the search space.

6. Experimental results

A simple branch-and-bound algorithm has been used for testing the model with several combinations of the functions to minimize using the weighted-sum method. Table 2 shows the number of nodes in the And/Or graphs corresponding to a set of hypothetical products of 30 and 40 components. Supposing that each individual component must be repaired, it includes the average number of Or, And (assembly tasks), And' (disassembly tasks) nodes, and of repair plans (#RP) in the simplified graphs. In order to obtain results about the behavior of the model for multi-mode tasks, each graph has been extended to include 10% multi-mode tasks (M30-1, M30-2, M30-3, M40-1, M40-2 and M40-3).

The CSP model described in this paper has been tested using a basic algorithm implemented in ILOG Solver [7]. A temporal limit of 300 seconds for single-mode and 600 seconds for 10% multi-mode has been established for the search. In order to guide the search, the order of selection of the variables to be instantiated is from up to down in the extended And/Or graph.

For this algorithm, 4 different objective functions of the repair plan have been selected to be minimized: the cost, the makespan, and two combined objective functions (applying the weighted-sum method for $w_t = 10$ and $w_c = 1$ (f_{MO10}) and for $w_t = 20$ and $w_c = 1$ (f_{MO20})). The values 10 and 20 have been chosen because the cost of a repair plan is, in average, around 15 times the total duration of the plan.

According to the 4 objective functions previously mentioned, some characteristics related to them have been studied: the quality of the solutions founded (table 3), the fraction of proven optimal solutions (table 4) and the execution time (table 5). Each row refers to a set of 80 instances of an And/Or graph for a hypothetical product of 30 or 40 components,with different combinations for the durations of tasks, machines and configurations used, and faulty component selected to be repaired. The cost of each task is a function depending on the machine, the configuration and the duration of this task. The experiments were carried out on a 2,66GHz Intel Core 2 Duo with 4GB RAM.

Regarding to the quality of the solutions (table 3), as expected, in most cases the cost function gets the best cost scores, the time one gets the best time scores, while the results obtained with f_{MO10} and f_{MO20} are between both of them. In some cases, such as 30-2 and M30-1, f_{Cost} gets the best score for both, time and cost. In table 4 it can be seen that the objective functions in which the cost has a high influence, get the best fractions of proven optimal solutions (f_{Cost} gets the best score, followed by f_{MO10}, f_{MO20} and, finally, f_{Time}) and, in most cases, also the fastest ones (table 5). This may be due to the differences between the costs of the tasks are more significant and discriminating than between the durations, and the cost seems to be less dependent of tasks order.

There are several important parameters that may be studied, such as the machine balancing or the parallelism in the repair plans. For example, regarding to the multi-mode problems, it has been founded a better solution (in time and cost) than for single-mode problems in 23,5% of the studied cases.

7. Conclusions and Future Work

This work proposes a CSP model for the optimal planning and sequencing of disconnection and connection tasks when repairing or substituting faulty components, taking into account the minimization of time and cost (multi-objective optimization). It is considered that the tasks can be executed in several operating modes. Some problems have been generated in order to test the proposed model.

As future work, it is intended to use different strategies to solve the problem working with heuristic algorithms based on the resulting state of the constraint propagation process and on the objective functions to be optimized. Furthermore, other approaches to solve multi-objective optimization problems are intended to be explored. Also, other objective functions can be considered.

Table 3. Comparative results (quality of solutions)

	f_{Time}		f_{MO20}		f_{MO10}		f_{Cost}	
Prob	T	C	T	C	T	C	T	C
30-1	393,77	5388,97	399,26	4958,79	406,17	4858,89	437,68	4744,15
30-2	984,85	14419,08	992,30	13970,64	974,20	13546,56	973,95	13203,27
30-3	1284,54	18043,72	1284,54	18043,72	1284,54	18043,72	1293,54	18041,87
40-1	849,13	11509,28	849,28	11496,69	849,28	11496,69	850,09	11491,81
40-2	407,05	5629,50	413,81	5302,51	422,35	5170,34	446,89	5080,22
40-3	701,06	9926,72	701,64	9897,00	703,87	9868,73	710,97	9578,25
M30-1	478,96	6690,59	520,46	6775,18	561,18	7076,17	456,30	4925,41
M30-2	996,60	14545,02	1005,31	14279,66	1005,79	14140,45	964,37	12952,39
M30-3	1256,96	17661,42	1257,68	17636,16	1257,87	17633,71	1260,81	17625,12
M40-1	922,60	12758,98	924,73	12614,76	925,26	12606,37	927,81	12592,89
M40-2	419,87	6135,54	421,42	5528,64	432,03	5530,75	489,17	5353,04
M40-3	667,32	9428,05	660,18	9199,45	671,73	9314,45	678,28	9082,78

Table 4. Fraction of Optimal solutions

Prob	f_{Time}	f_{MO20}	f_{MO10}	f_{Cost}
30-1	0,5	0,591	0,898	1
30-2	0	0,011	0,307	0,5
30-3	0	0	0	0
40-1	0	0	0	0
40-2	0,5	0,5	0,5	0,579
40-3	0	0	0	0,045
M30-1	0,5	0,5	0,5	0,954
M30-2	0	0	0,079	0,5
M30-3	0	0	0	0
M40-1	0	0	0	0
M40-2	0,397	0,5	0,5	0,557
M40-3	0	0	0	0,023

Table 5. Execution time (average)

Prob	f_{Time}	f_{MO20}	f_{MO10}	f_{Cost}
30-1	155,2	144,8	152,5	150,3
30-2	300	299,5	300	300
30-3	300	300	300	300
40-1	300	300	300	300
40-2	153,2	151,4	300	300
40-3	300	300	300	300
M30-1	420	368,5	349	121,9
M30-2	600	600	595,2	351,4
M30-3	540,4	600	600	600
M40-1	600	600	600	600
M40-2	416,5	318,25	307	270,4
M40-3	600	593,18	600	587,2

Acknowledgements

This work has been partially supported by the Spanish Ministerio de Educación y Ciencia through a coordinated research project (Grant DIP2006-15476-C02-01) and Feder (ERDF).

References

[1] M. Boddy, A. Cesta, and S. Smith. *ICAPS-04 Workshop on Integrating Planning into Scheduling.* AAAI Press, 2004.

[2] C. Coello. Evolutionary multi-objective optimization: A historical view of the field. *IEEE Computational Intelligence Magazine*, 1(1):28–36, 2006.

[3] K. Deb. Introduction to evolutionary multiobjective optimization. *5252 LNCS*, pages 59–96, 2008.

[4] C. Del Valle, A. Márquez, and I. Barba. A csp model for simple non-reversible and parallel repair plans. *Journal of Intelligent Manufacturing. To appear (DOI 10.1007/s10845-008-0162-9)*, 2009.

[5] Y. Haimes, L. Lasdon, and D. Wismer. On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE Transactions on Systems, Man, and Cybernetics*, 1:296–297, 1971.

[6] L. Homem de Mello and A. Sanderson. And/or graph representation of assembly plans. *IEEE Transactions on Robotics and Automation*, 6(2):188–189, 1990.

[7] ILOG. http://www.ilog.fr/.

[8] H. Ishibuchi and T. Murata. A multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Trans. Systems, Man and Cybernetics Part C: Applications and Reviews*, 28(3):392–403, 1998.

[9] R. Kolisch and A. Drexl. Local for multi-mode resource-constrained project. *IIE Transactions (Institute of Industrial Engineers)*, 29(11):987–999, 1999.

[10] A. Lambert. Disassembly sequencing: a survey. *Int. Jour. Production Research*, 41(16):3721–3759, 2003.

[11] W. Li and C. Zhang. Design for disassembly analysis for environmentally conscious design and manufacturing. *In Proc. 1995 ASME Int. Mechanical Eng. Congress*, 2:969–976, 1995.

[12] F. Rossi, P. Van Beek, and T. Walsh. *Handbook of Constraint Programming*. Elsevier, 2006.

[13] D. Smith, J. Frank, and A. Jónsson. Bridging the gap between planning and scheduling. *Knowledge Engineering Review*, 15(1):47–83, 2000.