How to take into account general and contextual knowledge for interactive aiding design: Towards the coupling of *CSP* and *CBR* approaches ☆

Élise Vareilles [a,*], Michel Aldanondo [a], Aurélien Codet de Boisse [a,b], Thierry Coudert [b], Paul Gaborit [a], Laurent Geneste [b]

[a] *Université deToulouse, École des mines d'Albi Centre de Génie Industriel, Route de Teillet Campus Jarlard, 81013 Albi Cedex 09, France*
[b] *Université de Toulouse, École national d'ingénieur de Tarbes, 47 av. d'Azereix, BP 1629, 65016 Tarbes Cedex 09, France*

Keywords:
Aiding design
Constraints satisfaction problem
Case-based reasoning
Case-based filtering
Helicopters maintenance

A B S T R A C T

The goal of this paper is to show how it is possible to support design decisions with two different tools relying on two kinds of knowledge: case-based reasoning operating with contextual knowledge embodied in past cases and constraint filtering that operates with general knowledge formalized using constraints. Our goals are, firstly to make an overview of existing works that analyses the various ways to associate these two kinds of aiding tools essentially in a sequential way. Secondly, we propose an approach that allows us to use them simultaneously in order to assist design decisions with these two kinds of knowledge. The paper is organized as follows. In the first section, we define the goal of the paper and recall the background of case-based reasoning and constraint filtering. In the second section, the industrial problem which led us to consider these two kinds of knowledge is presented. In the third section, an overview of the various possibilities of using these two aiding decision tools in a sequential way is drawn up. In the fourth section, we propose an approach that allows us to use both aiding decision tools in a simultaneous and iterative way according to the availability of knowledge. An example dealing with helicopter maintenance illustrates our proposals.

1. Introduction

Very often, designers start a new design from a previously studied situation that they could consider as a basis, and then, they finish the design task by using less contextual knowledge as rules or best practices, to adjust the solution to the requirements (Minsky, 1974). When the design domain is well known and design activity very routine, designers start with some kind of generic solution that they instantiate according to the requirements, while comparing the result with previous designs. Therefore, there is no doubt that, most of the time, designers take into account two kinds of knowledge simultaneously: contextual knowledge corresponding to past cases and general knowledge corresponding to relations, rules or constraints that link design variables.

If the designers are able to handle and use these two kinds of knowledge, aiding design tools should be able to do the same and process contextual and general knowledge. As very few studies have taken an interest in this kind of knowledge coupling, the goal of this paper is to show that it is possible to support design decisions with two different tools relying on these two kinds of knowledge. We will consider on one hand, a constraint filtering tool working with general knowledge formalized as a constraint satisfaction problem (*CSP*), and secondly, a case-based reasoning tool (*CBR*) operating with contextual knowledge.

Some studies have shown that approaches like *CSP* and *CBR* are good candidates to assist design decision, for example Dutta et al. (1997), Nemati et al. (2002), Fargier et al. (1996) or Goel and Craw (2006). In this paper, we study how they can cooperate in order to better support design. This cooperation, or coupling, is based on the complementarity of general and contextual knowledge. Therefore, we propose a new way of combining these two types of knowledge in order to take the most of them: they are not just used in sequence, one feeding the other, as it can be found in the literature and presented on the left part of Fig. 1, but in a real complementary way by exchanging knowledge, as shown on the right part of Fig. 1.

Consequently, the paper is organized as follows. In the remainder of this section, we recall the background of constraint filtering (cf. Section 1.1) and case-based reasoning (cf. Section 1.2) and exhibit the context of our proposals (cf. Section 1.3). In Section 2, an example which comes from the industrial problem and runs throughout the paper is presented. In Section 3, we describe briefly and illustrate the various possibilities for associating the two aiding design tools (*CSP* and *CBR*) which have been found in the

☆This article is an extended version of a communication presented at MOSIM 2010.

* Corresponding author.
E-mail address: elise.vraeilles@mines-albi.fr (É. Vareilles).

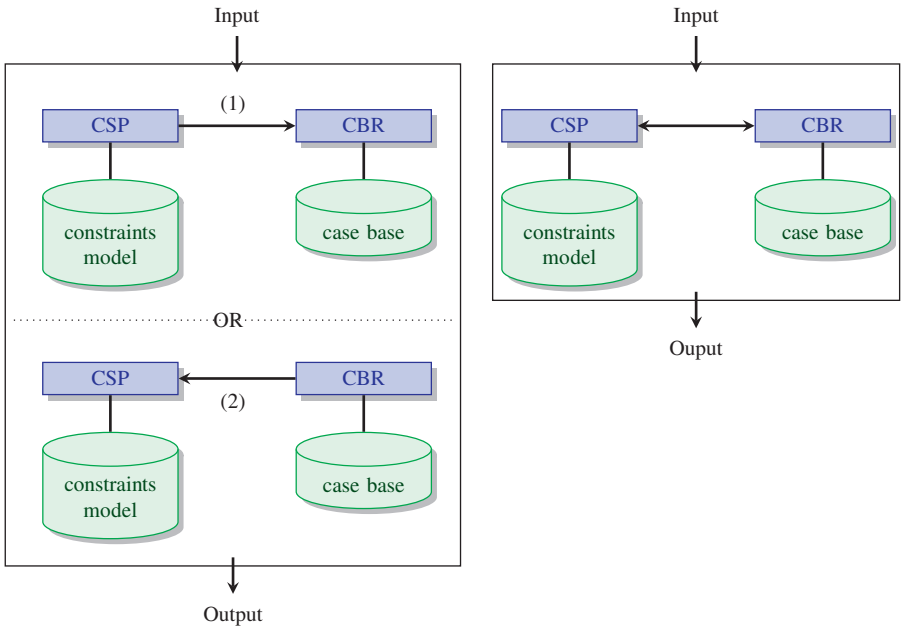

Fig. 1. Sequential and simultaneous coupling of *CBR* and *CSP*.

literature. These associations are mainly done in a sequential way, meaning that one of the tools is used first to supply the second one in order to reach a solution. In Section 4, we present our approach that enables us to use both aiding decision tools in an interactive and simultaneous way, meaning that the tools have to share and exchange knowledge in order to reach a solution.

1.1. CSP approaches

When dealing with constraint filtering to assist design, knowledge is explicitly expressed as constraints linking the variables of the problem (Chenouard et al., 2009). A constraint can take various forms: lists of allowed combinations such as $\{(x=1,y=a),(x=2,y=b)\}$, mathematical formulae such as $y=x^2+3-x$, or logical rules, such as $(A\vee B)\wedge C$. All the constraints are gathered in a model which corresponds to the knowledge, expressing what can be accepted or forbidden. In order to find a solution interactively, the user defines her/his requirements on some of the model variables. The reasoning process consists in reflecting these requirements through the constraints network to the other variables by limiting their domains only to consistent values. This mechanism, repeated several times, restricts the solution space progressively to reach consistent solutions.

More formally, a *CSP* can be defined by a triplet $\{X,D,C\}$. X is a set of variables, D is a set of domains (one domain for each variable) and C is a set of constraints where a constraint defines the allowed or forbidden combinations of variable values. *CSP* filtering techniques allow decision results to be propagated interactively on a network of constraints. This kind of decision propagation is also similar to the human ability to deduce consequences from facts. Many works that consider aiding design as a constraint satisfaction problem have been achieved, for example White et al. (2009), Bin et al. (2010), Bodirsky and Dalmau (2006) or Vareilles et al. (2007).

1.2. CBR approaches

In case-based reasoning or *CBR* (Riesbeck and Shank, 1989; Aamodt and Plaza, 1994), systems expertise is embodied in a library of past cases, rather than being encoded in classical rules. Each case typically contains a description of the problem, plus a solution and/or the outcome. The knowledge and reasoning process used by designers to solve the problem is not recorded, but implicit in the solution. In order to find a solution, the user describes her/his problem through a list of variables and after all user inputs, the described problem is matched against the cases in the past case base. A similarity function (Kolodner, 1993) let you detect and classify the similar past cases and the most similar ones are retrieved. If the user's problem does not match against any past cases, the system will return the nearest possible ones. The retrieved cases provide ballpark solutions that, generally, must be adapted by the user to fit her/his current problem. Once revised by an expert, the case is added to the case base.

More formally, a case-based reasoning problem can be defined by a triplet $\{A,D,F\}$. A is a set of attributes or variables that are used to describe the case, D is a set of distances (one distance for each attribute) that permits us to calculate a measurement between the values of each attribute and F is an operator that aggregates the distances into a single similarity measurement in order to have the similarity score of each case. *CBR* techniques support analogy reasoning which is very frequently used by humans for solving problems. Consequently, *CBR* has been used in numerous works dealing with aiding design, for example Althoff (2008), Changchien and Lin (2005), Bergmann et al. (2003), Carsten (2001) or Bichindaritz and Marling (2006).

1.3. Proposals context and background

Our proposal is therefore to analyse the association of the two aiding design tools, *CSP* and *CBR*, in order to use them on a same design problem. According to Brown and Chandrasekaran (1984) and Coyne et al. (1989), product design can be characterized with respect to a degree of recurrence in: creative, innovative and routine design. They have also outlined the kind of variables and knowledge necessary to achieve these three kinds of design. As knowledge availability is a strong prerequisite for these two knowledge based tools, we only consider routine design situations.

Therefore, the variables describing design requirements and design solutions, which we shall henceforth call design problem variables, noted *Vdp_i*, will be considered by the two aiding design

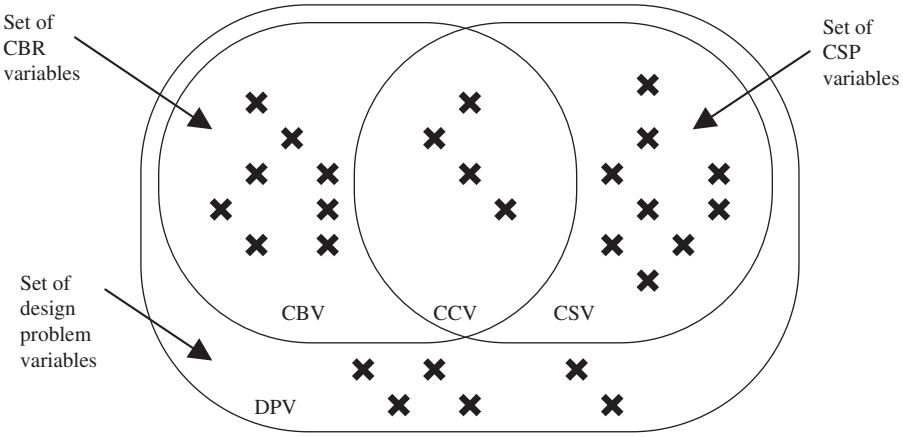

Fig. 2. Three sets of design problem variables.

tools. As general (*CSP* used) and contextual (*CBR* used) knowledge do not necessarily act on the same subset of design problem variables, we can consider three variable sets as represented in Fig. 2:

- one set of design problem variables, noted $DPV = \{Vdp\_i\}$,
- two subsets of *DPV*, where each variable is associated with:
  - a *CSP* variable, noted $CSV = \{Vcs\_k\}$.
  - a *CBR* variable, noted $CBV = \{Vcb\_j\}$,

Most of the time, the set of variables *DPV* is much greater than each of the two subsets, because it is quite difficult to include all design variables in a knowledge model. When comparing the two subsets *CSV* and *CBV*, we consider that they are different but have some design variables in common $CCV = CSV \cap CBV \neq \emptyset$, which means that contextual and general knowledge cover different design problem variables. Thus the knowledge of the two aiding design tools is quite different and can be complementary.

Our goal is then to establish some kind of cooperation of the two kinds of knowledge with the two aiding design tools. A new way of combining these two types of knowledge by exploiting their complementarity is then proposed and detailed in Section 4: the *CSP* and *CBR* tools work together by exchanging knowledge in order to find a solution.

2. Industrial problem: needs and example

Previous aiding design tools have been initially proposed for product design. As long as the design process is rather routine, they can be used for any kind of artefact. As a consequence, they are now widely used for aiding the design of systems, processes or services. The example that illustrates this paper deals with the design of maintenance processes for helicopters.

According to Norme NF X60012 (2006), maintenance means any of the operations required to maintain or re-establish a product in a specified state or to guarantee a predetermined service. In the aeronautical field, average maintenance costs during the life of an aircraft are higher than the initial purchase costs for the products. We must notice that helicopter maintenance cost represents 45% of the overall life cycle cost (Poncelin et al., 2006).

In order to manage the helicopter maintenance process, it is essential for each maintenance job firstly to be able to define maintenance operations accurately, and secondly to match the work load with the required resources, in order to estimate the maintenance cost for each job associated with a customer demand. In order to do so, maintenance managers operate with two knowledge sources: the helicopter technical documentation, which can be seen as general knowledge and their own past experience corresponding with previous maintenance jobs which can be seen as contextual knowledge. By maintenance job, we are referring to a full maintenance service between helicopter landings and take offs. Three types of maintenance services exist:

(i) complete service *CS*: in such a case, the helicopter is completely dismantled and all of the parts are tested and replaced if necessary,
(ii) interim service *IS*: in such a case, only the critical parts are tested and changed if necessary (blades, rotor, for example), and
(iii) mini service *MS*: in such a case, the less critical parts are tested and replaced if necessary (air-conditioning, seats or oil changes for example).

Helicopter technical documentation, given by the constructors, is called the *Maintenance Report Board* or *MRB* and must be followed absolutely to the letter. The *MRB* defines the cycles of maintenance for a family of helicopters and each elementary operation (time interval between maintenance operations, the type of maintenance operation, required resources). Globally the *MRB* defines what should be done in order to achieve smooth and safe helicopter behavior. Therefore *MRB* defines all required operations and required resources for each maintenance job. Maintenance workload and cost can be consequently quantified by experts in maintenance for each maintenance operation and aggregated for the maintenance job.

However, if the *MRB* explains what must be done, most of the time many other operations must be added, according to the effective utilization of the helicopter. These operations are non-critical for security but most of the time they prevent early return and improve the helicopter availability. In fact when the helicopter lands for a maintenance service, a detailed diagnosis operation is carried out in order to update the set of maintenance operations. Thus, these added operations modify the initially estimated workload and cost and can generate maintenance delay and loss of control of the maintenance process. The information related to these added operations (including workload and cost updates) corresponds in fact with the contextual knowledge and is stored in the case base. The idea is to use this knowledge to establish, for each maintenance job, more accurate maintenance operation plans, workload and cost estimations.

2.1. General knowledge handled by the CSP model

Given previous elements, the *MRB* is considered as general knowledge and can be formalized as a constraint satisfaction problem defined as follows.

2.1.1. Definition of the maintenance job

In order to define all maintenance operations, the *MRB* requires at least the five following variables, prefixed with the *MRB_JOB* symbol:

- Helicopter type: corresponds to the helicopter model. There are four models in our example that are modelled using a symbolic variable: *MRB_JOB_HT* with a symbolic domain: {*Dragon*, *Puma*, *Tiger*, *Dolphin*}.
- Maintenance job type: corresponds with three kinds of services. There are modelled using a symbolic variable *MRB_JOB_ST* with a symbolic domain: {*complete*, *interim*, *mini*}.
- Helicopter age: corresponds to the number of years since first take off. This age is modelled using a numerical variable *MRB_JOB_HA* with a continuous domain [0, 100].
- Helicopter flight hours: corresponds to the number of flight hours since first take off. This variable, named *MRB_JOB_FH*, has the following continuous domain [0, 10 000].
- Helicopter equipment: corresponds with five kinds of utilization that require specific equipment. They are modelled using a symbolic variable *MRB_JOB_HE* with a symbolic domain {*Standard*, *Rescue*, *Camera*, *Agriculture*, *Military*}.

2.1.2. Definition of the maintenance operations

The variables describing these operations are prefixed by *MRB_OPR*. Four operations always exist for any maintenance job:

- diagnosis, first operation noted:*MRB_OPR_Diagnosis*;
- disassembly, second operation noted: *MRB_OPR_Disassembly*;
- assembly, before last operation noted: *MRB_OPR_Assembly*;
- final test, last operation noted: *MRB_OPR_Finaltest*.

All other operations are between previous operations and are characterized with:

- A helicopter part *HP* that corresponds to the part of the helicopter where the operation takes place: {*Structure*, *Engine*, *Transmission*, *Cabin*}.
- A technology required *TC* that corresponds to the main technology of the operation: {*Mechanical*, *Electrical*, *Computer*, *Hydraulics*}.

Therefore, 16 operations (four parts and four technologies) can be present between disassembly and assembly. These 16 operations are noted: *MRB_OPR_HP-TC*.

These 20 operations, noted *MRB_OPR_i* with $i=1$–20 are defined by three operation description variables, prefixed with the *MRB_OPR_i* symbol:

- An operation level that corresponds to four levels of complexity and a none level meaning that the operation is not required for this maintenance job. This operation level is modelled using a symbolic variable *MRB_OPR_i_LV* with a symbolic domain {*very_complex*, *complex*, *light*, *very_light*, *none*}.
- An operation resource that corresponds to the main technical resource that is required. This resource is modelled using a symbolic variable *MRB_OPR_i_TR* with a symbolic domain {*tr*_1,*tr*_2, . . . ,*tr*_50}.
- An operation competency that corresponds to the main human skill that is required. This skill is modelled using a symbolic variable *MRB_OPR_i_HC* with the symbolic domain {*hc*_1, *hc*_2, . . . ,*hc*_50}.

Given these elements, the *CSP* model brings together two subsets:

- of five variables describing the maintenance job: *MRB_JOB_V*,
- of $60=(20*3)$ variables describing the maintenance operations: *MRB_OPR_V*.

The knowledge of the *MRB* permits us to identify and model the constraints between the two previous variable subsets *MRB_JOB_V* and *MRB_OPR_V*. These constraints express the allowed combinations of variable values.

In order to quantify workload and cost, the maintenance experts, relying on their own specific knowledge, have derived workload and cost from *MRB* for each maintenance operation:

- Operation workload: corresponds with the amount of *man∗hours* required by the operation and is modelled using the variable *OPR_i_WLD*-1.
- Operation cost: corresponds with the standard cost in euros of the operation and is modelled using the variable *OPR_i_CST*-1.

and the constraints that link these two indicators (for each operation) with previous variables belonging to *MRB_JOB_V* and *MRB_OPR_V*. The 40 ($=20*2$) variables corresponding with these two indicators belong to a variable set called *OPR_WLD-CST_V*. The full *CSP* model gathers 105 variables. Its architecture is shown in the right part of Fig. 3.

2.2. Contextual knowledge handled by the CBR model

The content of contextual knowledge is directly associated with the effective maintenance process. Therefore each maintenance job and each of its maintenance operation are characterized by:

- Two variables corresponding with previous indicators and defined in the *CBR* variable list:
  - Operation workload: corresponds to the amount of *man∗hours* effectively used. This workload is modelled using the variable *OPR_i_WLD-EFF*.
  - Operation cost: corresponds to the effective cost of the operation. This cost is modelled using the variable *OPR_i_CST-EFF*.
- Three operation description variables that corresponds to the effective maintenance operation:
  - Operation level: corresponds to the effective operation level. This level is modelled using the symbolic variable *OPR_i_LV-EFF* with the symbolic domain {*Very_complex*, *complex*, *light*, *very_light*, *none*}.
  - Operation resource: corresponds with the effective main technical used resource. This kind of resource is modelled using the symbolic variable *OPR_i_TR-EFF* with the symbolic domain {*tr*_1,*tr*_2, . . . ,*tr*_50}.
  - Operation competency: corresponds to the effective main human competency used. This kind of resource is modelled using the symbolic variable *OPR_i_HC-EFF* with the symbolic domain {*hc*_1,*hc*_2, . . . ,*hc*_50}.

These variables are included in the two following variable sets:

- Of 40 variables corresponding to effective workload and cost: $EFF\_OPR\_WLD{-}CST\_V = \{OPR\_i\_WLD{-}EFF\} \cup \{OPR\_i\_CST{-}EFF\}$.
- Of 60 variables corresponding to effective maintenance operations description: *EFF-OPR_V*={*OPR_i-EFF*}.

The differences between effective values belonging to {*EFF-OPR_V*} and {*EFF_OPR_WLD-CST_V*} and initially provisional values belonging to {*MRB-OPR_V*} and {*OPR_WLD-CST_V*} result from helicopter effective conditions of utilization. If the helicopter is lightly used or in very good condition, the difference between provisional and effective values is rather small. This is of course not the case when the helicopter is used in very severe conditions.

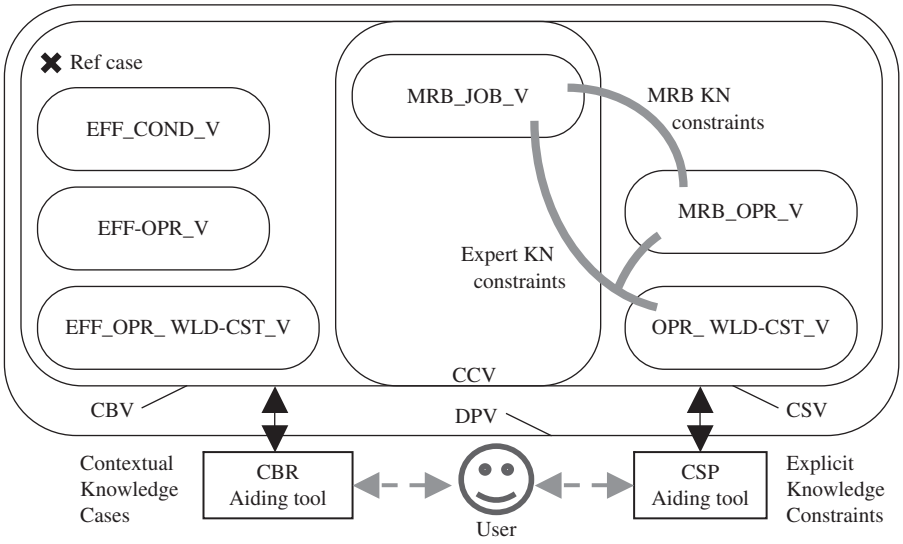

Fig. 3. *CBR* and *CSP* knowledge models of the example.

In that case many maintenance operations have to be added, as explained earlier.

Effective conditions of utilization are characterized by the following four variables prefixed by *EFF_COND* symbol:

- Aggressive environment: corresponds to specific environments that have very strong consequences on the helicopter availability such as: sea/salt, desert/sand, mountain/snow. These environments are modelled using the variable *EFF_COND_ENV* with the symbolic domain {*normal*, *sea*, *desert*, *mountain*}.
- General use: characterizes the utilization severity. This use is modelled using the symbolic variable *EFF_COND_SEV* with the symbolic domain {*very_severe*, *severe*, *normal*, *light*, *very_light*}.
- Heavy load: characterizes the load level that has a strong impact on engine and transmission. This characteristic is modelled using the variable *EFF_COND_LOD* with the symbolic domain {*very high*, *high*, *normal*, *low*, *very low*}.
- General care: characterizes if various pilots or owners take more or less care of the helicopter. This owner's characteristic is modelled using the variable *EFF_COND_CAR* with the symbolic domain {*low care*, *normal*, *great care*}.

These four variables belong to the variable set *EFF_COND_V*.

In order to identify the maintenance job, the *CBR* case model therefore contains:

- a reference number of the helicopter visit,
- five variables describing the maintenance job *MRB_JOB_V*,
- four variables for use condition characterization related to the job *EFF_COND_V*,
- sixty variables for effective maintenance operation description *EFF-OPR_V*,
- forty variables for effective workload and cost of each operation to *EFF_OPR_WLD-CST_V*.

The complete *CBR* model is composed of 110 variables. Its architecture is shown in the left part of Fig. 3.

2.3. Architecture of knowledge model and synthesis

Given previous elements, it is possible now to assemble the two knowledge models as shown in Fig. 3. The complete knowledge model brings together the following variable subsets

- for *CSP* and *CBR* models:
  - *MRB_JOB_V* job description (five variables), right in the middle of Fig. 3,
- for *CSP* model, on the right part of Fig. 3:
  - MRB_*OPR_V*: maintenance operation description ($20 \times 3 = 60$ variables),
  - constraint between MRB_*JOB_V* and MRB_*OPR_V*, grey line in Fig. 3 corresponds to *MRB* knowledge,
  - *OPR_WLD-CST_V*: operation workload and cost ($20 \times 2 = 40$ variables),
  - constraint between *MRB_JOB_V*, *MRB_OPR_V* and *OPR_WLD-CST_V*, grey line in Fig. 3 corresponds to maintenance expert knowledge,
  - constraint between *MRB_JOB_V* and *MRB_OPR_V*, grey line in Fig. 3 corresponds to *MRB* knowledge,
- for *CBR* model, on the left part of Fig. 3:
  - reference number of the helicopter visit (Ref case),
  - *EFF_COND_V*: use condition (four variables),
  - *EFF_OPR_V*: effective maintenance operation description ($20 \times 3 = 60$ variables),
  - *EFF_OPR_WLD-CST_V*: effective workload and cost ($20 \times 2 = 40$ variables).

The complete model is then composed of 210 design problem variables *DPV*. This example has clearly highlighted the needs of taking into account different sources of knowledge in order to make better decision. In this particular context of helicopter maintenance, general knowledge (*MRB*) and contextual knowledge (past cases) have to be considered at the same time for estimating the time to carry the service, quoting the more precisely the maintenance price, dealing with any possible risks caused by the operations itself and more importantly, avoiding keeping the helicopter out of service any longer than necessary.

3. Association of *CSP* and *CBR* based aiding design tool: related work

For this section, we consider that the two aiding tools are composed of a knowledge base and a processing unit. For the *CSP* tool, the knowledge base is the constraint model and the processing unit is the constraint filtering treatment. For the *CBR* tool, the knowledge base is the case base and we consider for the

processing unit only the retrieval; we consider the adaptation, revision and retention as human process.

We assume in this paper that the knowledge models are inputted, updated and validated by experts while the processing units interact with a user that has a design problem to solve. This means that we assume that the actor, whom we call the user, cannot input a new case in the case base, because we consider that updating knowledge should be achieved by the actor we call the expert. Fig. 4 represents the knowledge base and processing unit of each aiding tool with relevant users and experts that interact with them.

We can see in Fig. 4 that each tool can interact:

- at the knowledge level with a knowledge expert:
  - inputting or outputting knowledge:
    - model, piece of model, constraints or piece of constraints for *CSP*, this knowledge results from periodic knowledge extraction relying on expert judgment and past cases and experience,
    - group of cases, case or piece of case for *CBR*, this knowledge results from effective maintenance operations records that are considered as adequate and consistent for the case base.
- at the problem level with a user:
  - inputting piece of problem or outputting a piece of solution:
    - entering variable values for *CSP* or *CBR*,
    - getting variable domain restrictions from *CSP* or *CBR*.

The goal of this section is to analyse various combinations of input/output, knowledge base/processing unit of the two aiding tools. Some associations do not present any interest while others correspond to already studied ideas, for an overview of ancient works done before 1995 see Sqalli et al. (1999).

The association overview is organized in three sub-sections dealing first with knowledge validation (Section 3.1), then with knowledge enrichment (Section 3.2) and finally with sequential use of the two aiding tools (Section 3.3).

3.1. Knowledge validation

By validation, we mean that one of the two knowledge bases (or a part of one) can be used to validate the other one (or a part of it). This assumes that the variables supporting knowledge are present in both knowledge models (*CSP* and *CBR*) and therefore belong to the intersection *CSV* and *CBV*. This also means that the experts agree on the fact that they have more confidence in one of the two knowledge bases (or a part of one).

3.1.1. Validation of general knowledge base

For this validation, the higher confidence is in the contextual knowledge encapsulated in the cases stored in the *CBR*. The *CBR* cases are therefore used to validate or invalidate the knowledge of the constraint model. An idea, which has been proposed and discussed by Felfernig et al. (2007) is to take all cases of the case base and to input them sequentially in the constraint filtering process as shown in Fig. 5.

According to:

- the number of variable values that are missing in the constraint model,
- the number of constraints of the constraint model that are not respected,

it is possible to quantify for each case a quality grade that characterizes each variable domain validity and each constraint consistency. These atomic numbers can then be aggregated in order to provide a global consistency score for the whole constraint model. With these elements, the experts can decide to validate or invalidate the knowledge model or a part of it.

Illustrative example:

Situation 1: Initial situation: The helicopter maintenance problem is considered with the full knowledge model described in Fig. 3 in Section 2. We consider a fleet of similar helicopters belonging to a specific customer.

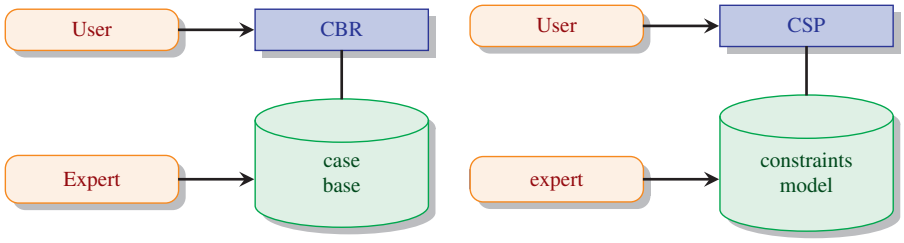

Fig. 4. Aiding tools user and knowledge expert.

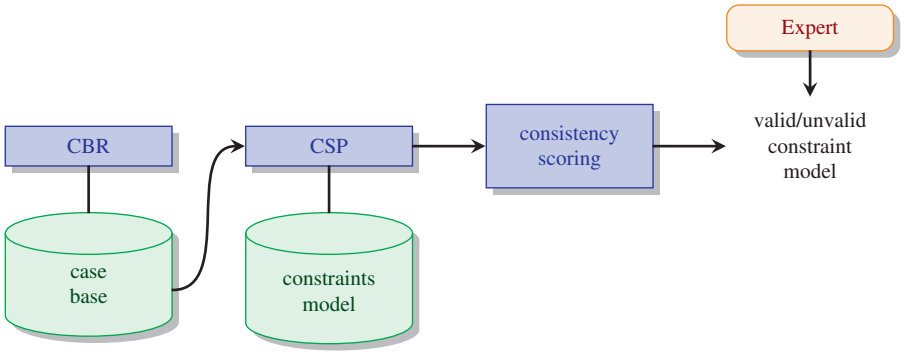

Fig. 5. Validating *CSP* knowledge base with *CBR* knowledge base.

This fleet has been maintained for many years always under the same kind of use conditions. Therefore many maintenance cases have been accumulated. As we assume a very similar use condition for all helicopters, experts in maintenance have extrapolated and established the knowledge to quantify workload and cost from *MRB* (expert KN constraints in Fig. 3) without taking into account use conditions (*EFF_COND_V*).

Situation 2: New situation: We assume now that the utilization conditions of this fleet are completely modified (modifications of: aggressive environment, severity utilization, helicopter load, general care). Therefore previous knowledge (expert KN constraints) or pieces of knowledge are not valid anymore. Thus, once a certain quantity of maintenance operations is achieved with the new utilization conditions and relevant cases stored in the case base, it is possible to check the constraint model. The result would indicate the constraints that are now invalid and have to be updated.

3.1.2. *Validation of contextual knowledge base*

In that case the confidence is in general knowledge embedded in the constraint model, and this *CSP* model is used to validate or invalidate cases corresponding to the *CBR* contextual knowledge. The previous validity checking process idea is considered again (Fig. 6), but the difference is that a consistency score is necessary in order to qualify the validity of each case. Therefore:

- the number of variable values that are missing in the case description,
- the number of constraints of the constraint model that are not respected,

allow us to quantify a case consistency score. According to this score, the experts can decide to valid or invalid the case.

Illustrative example:

Situation 1: Initial situation: The previous situation 1 described in Section 3.1.1 is again considered without any modification.

Situation 3: New situation: We assume now that the customer has bought other similar helicopters in order to increase his fleet size. Maintenance records are provided with these helicopters and should be added as cases in the case base. Thus, in order to guarantee the quality of the knowledge base of the *CBR*, each case should have its validity checked with respect to the constraint model. The result would indicate, for each case, if it can be used or not to update the *CBR* knowledge model.

3.2. *Knowledge enrichment*

By enrichment, we mean that one of the two knowledge bases (or a part of one) can be used to provide some knowledge to the other one. For this section we assume a high confidence in the source of knowledge. As in the previous sub-section, it is assumed that the variables supporting knowledge are present in both knowledge models (*CSP* and *CBR*) and therefore belong to the intersection $CCV = CBV \cap CSV$.

3.2.1. *Enrichment of general knowledge base*

In this case, the cases stored in the case base are used to generate knowledge for the *CSP* model. Such added knowledge can correspond with:

- adding a value in a variable domain,
- adding a variable in the constraint model,
- adding an allowed combination of values in a constraint,
- adding a constraint in the constraint model.

The first two modifications are rather obvious, because they can be deduced by a comparison of the structure of the two knowledge models (list of variables and list of values for each variable). The two others are more delicate because they rely on knowledge extraction and/or identification techniques (statistical regression, data analysis, data mining). This enrichment process is shown in Fig. 7.

Illustrative example:

Situation 1: Initial situation: The previous situation 1 described in Section 3.1.1 is again considered without any modification.

Situation 4: New situation: We assume now that the helicopter use conditions have changed. If use conditions were

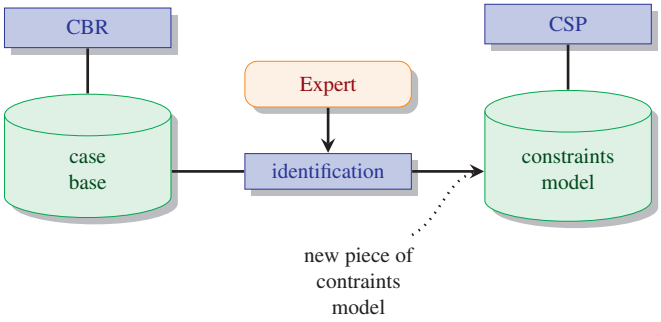

Fig. 7. Enrichment of *CSP* model with case analysis and identification.

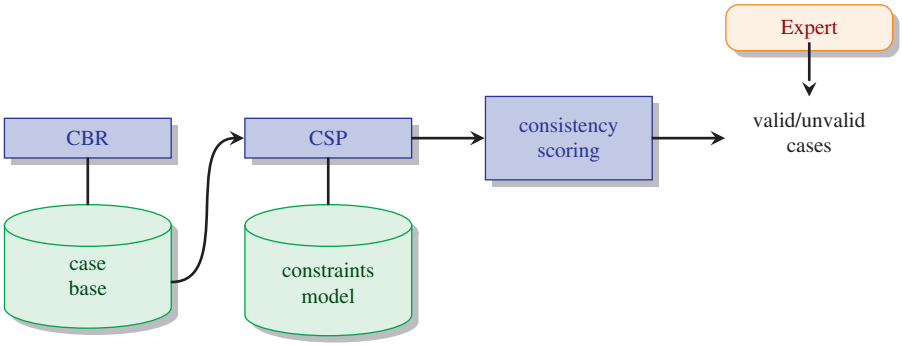

Fig. 6. Validating *CBR* knowledge base with *CSP* knowledge base.

initially roughly similar, they are now very diverse. After some years with these diverse use conditions, many helicopter maintenance jobs have been carried on and a large case base has been set up. In that situation, it appears interesting to try to add to the *CSP* model of each maintenance operation a couple of variables that could estimate work load and cost while taking into account utilization conditions. Thus, the provisional maintenance plan and cost could be much more accurate. In that case, it could be difficult either for an expert in maintenance to identify a rule linking the work load and the cost directly to the utilization use. For this reason, it is interesting to use, for instance, data analysis to highlight the link between these variables.

Thus the structure of the updated knowledge model represented in Fig. 8 shows:

- effective utilization condition variables (variable set *EFF_COND_V*) are now shared by the two *CBR* and *CSP* models and belongs now to *CCV*, moving from upper left to lower centre part of Fig. 8,
- for each maintenance operation, two estimated variables are added to *CSV* – workload and cost – which take into account effective utilization conditions (variable set *OPR_WLD-CST_V2*), lower right part of Fig. 8,
- constraints that allow us to compute these estimation variables are added, linking the variables sets: *OPR_WLD-CST_V*, *EFF_COND_V* and *OPR_WLD-CST_V2*. The definition of the last computation constraints is obtained with a regression analysis that processes all cases of the case base.

3.2.2. *Enrichment of contextual knowledge base*

In this case, the knowledge stored in the constraint model is used to generate knowledge for the *CBR* model. Most of the time this process can be used when a case is incomplete by missing at least one variable value. This occurs for example when data is lost during case collection or when a group of cases are imported and miss a variable value on all cases. This corresponds with some cases with "knowledge holes" that should be filled.

In order to obtain the missing value, each case is processed by the constraint filtering unit in order to propose a value for the required variable. Once the case missing value is obtained, it is inputted in the case base and the contextual knowledge is updated as shown in Fig. 9.

Illustrative example:

Situation 1: Initial situation: The previous situation 1 described in Section 3.1.1 is again considered without any modification.

Situation 5: New situation—-situation 3 of Section 3.1.2 is again considered. In this situation similar helicopters are bought and their maintenance records must be added as cases in the case base. The difference is that now we assume that the records are incomplete with respect to the case model (some values are missing for certain variables). These knowledge holes can be filled thanks to constraint propagation and cases relevant to the acquired helicopters can be inputted in the case base.

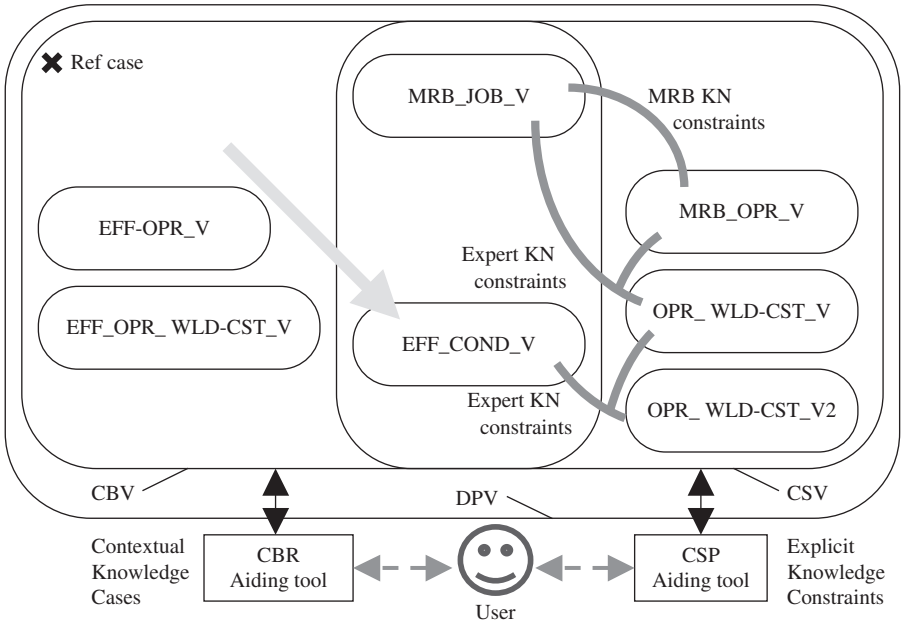

Fig. 8. Model example.

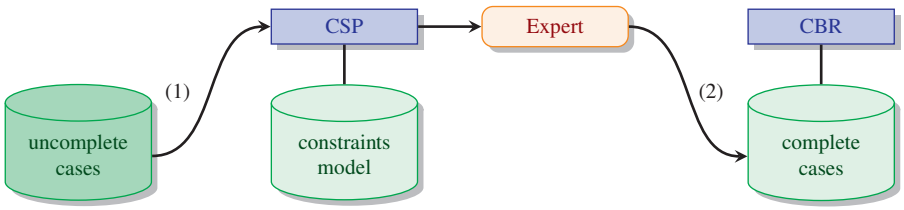

Fig. 9. Enrichment of *CBR* model with constraint filtering.

3.3. Sequential use of aiding tools

Until now, we have only dealt with knowledge validation or completion. We have not dealt with knowledge processing with *CBR* and *CSP* in order to assist design. This section is therefore concerned by the sequential use of the two tools, meaning the output of one is the input of the other without any iteration. Two kinds of behaviours have been identified:

- the second tool is used to assist the end of the design task mainly accomplished by the first one,
- the first tool is used to assist preparation of the design task mainly accomplished by the second one.

3.3.1. Aiding decision with CBR then CSP

In this situation shown in Fig. 10, the case base system is used firstly to retrieve a similar case and secondly, the constraint propagation is used to adapt the case. This method of proceeding is very interesting when constraint models are valid only on a small part of the solution space. In that case *CBR* can roughly identify a part of the solution space that matches the customer's requirements. Then, the appropriate constraint model (the one defined for the previous part of the solution space) can be selected and used in order to adapt the case and thus to terminate design. This association of tools has been studied by some authors, such as Purvis and Pu (1995), Inakoshi et al. (2001), Ruet and Geneste (2002), Lopez (2003) and Roldan et al. (2010).

Illustrative example:

Situation 1: Initial situation: The previous situation 1 described in Section 3.1.1 is again considered without any modification.

Situation 6: Not a new situation—There has not a new situation (because knowledge in not changed) but the operating mode is to mainly use the *CBR* and, if matching is not perfect, to adapt a part of the case with constraint filtering (computer technology maintenance operation in our example). Given this, the design process could consist sequentially in:

- using case retrieval in order to identify the case with most similar job description (*MRB_JOB_V* variables) and consider all effective maintenance operation description (*EFF_OPR_V* variables) and effective workload and cost (*EFF_OPR_WLDCST_V*) from the previous selected case,
- if the selected case does not match exactly the subset of variables (helicopter type, helicopter equipment, helicopter age), adapting the case by replacing all effective maintenance operation descriptions (*EFF_OPR_V*) dealing with computer technology by *MRB* maintenance operation description (*MRB_OPR_V*). Globally, this means that for computer technology, the *MRB* knowledge must be considered instead of the case knowledge.

3.3.2. Aiding decision with CSP then CBR

In this situation shown in Fig. 11, the filtering system is used in order to assist the definition of the input of the *CBR*. This way to proceed can be interesting either when the case structure presents a large number of variables or when some constraints express well-known knowledge that cannot be disputed.

We can note that Sqalli and Freuder (1998) have proposed the use *CSP* first then *CBR* if the *CSP* model is incomplete or incorrect and fails to solve the problem. The *CBR* checks if there is a similar case in the case-base from the *CSP* values in order to adapt its solution and solve the current problem. Our proposals lies within these works in which the knowledge bases are incomplete but consistent.

Given the above, the user inputs each requirement in the constraint filtering unit. After each input, constraints filtering reduces the domain of other variables. Once this process is completed:

- it is possible to propose a set of requirements to the *CBR* that is consistent with the constraints, therefore the quality of the retrieval process is greatly improved,
- as the number of variables of the *CBR* model is, most of the time, larger than the *CSP* model, the user can complete the requirements set for the *CBR*.

Thus, the inputs of the *CBR* are set with a better quality (consistent with the constraint model) and are quicker (constraints reduce the input possibilities).

Illustrative example:

Situation 1: Initial situation: The previous situation 1 described in Section 3.1.1 is again considered without any modification.

Situation 7: Not a new situation—As before, knowledge is the same and the difference lies in the way to use the aiding tools. The goal of using *CSP* is therefore to

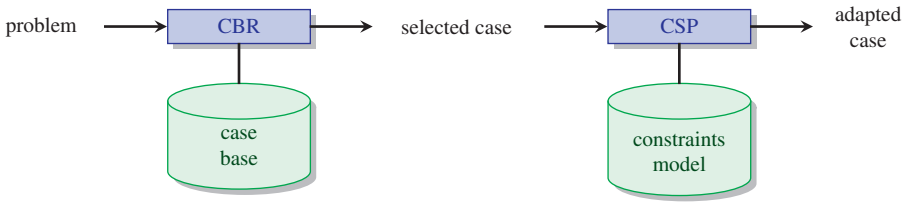

Fig. 10. Aiding decision with *CBR* followed by a *CSP* adaptation.

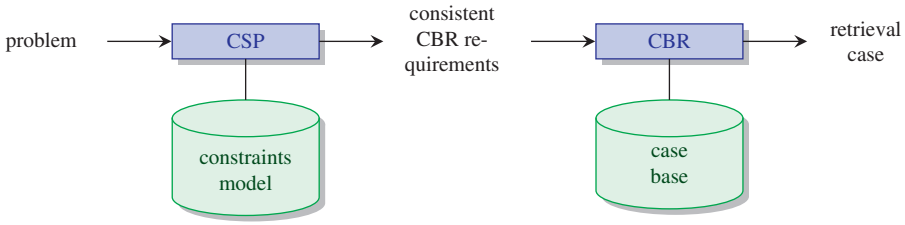

Fig. 11. Aiding decision with *CBR* pre-processed by a *CSP*.

prevent inconsistent input to the *CBR*. We consider an inconsistent input, for example that a *Dolphin* helicopter type cannot receive the helicopter equipment *military* or *rescue*. A *CBR* does not know what constitutes an inconsistent input and will always provide a response. The goal of the present operating mode is to avoid searching on an inconsistent input as follows:

- each variable value describing the problem to solve is first processed by the constraint filtering unit,
- in our example the *dolphin* selection for the helicopter type would have pruned *military* or *rescue* from the helicopter equipment list and forbidden the inconsistent input,
- furthermore, the pruning of other variable values would provide a better quality input and guide the search of the *CBR*.

3.4. Synthesis

The goal of this section has been to analyse, thanks to related works, various ways to combine case retrieval and constraint processing for aiding design.

In the first two sections, we dealt with knowledge validation and knowledge completion. They do not directly assist the user in the sense that they do not provide any advice or assistance for aiding design. But when the problems of knowledge maintenance, evolution and validation are addressed, the various ways to associate these two concurrent approaches allow us to improve confidence in knowledge and thus in the two aiding design tools.

In the last section, we have dealt with knowledge utilization for aiding design. The two proposed operating modes gathering case retrieval and constraint processing can clearly assist the designer. But as the proposed operating modes use these two tools in a sequence, they cannot combine the two kinds of knowledge on a same design problem as the designer might expect. In order to achieve a more effective cooperation, in the next section we present a more complex association of the two knowledge processing tools in order to be able to combine simultaneously the two different types of knowledge on a same design problem.

Our approach to this integration is novel: firstly, the two tools work together in a simultaneous way by exchanging and sharing knowledge in order to reach a solution, contrary to what can be found in the literature. Secondly, they are used in an interactive way, meaning that the solution space is progressively reduced by the inputs of the designer.

4. Proposal for interactive coupling process

In this section, we propose an interactive way of using general and contextual knowledge at the same time in order to help designers make better design decisions. We make the same assumption that we did at the beginning, a general knowledge base modelled as a *CSP* and a contextual knowledge base model as a *CBR*.

It must be stressed that the interactive coupling process is invisible for the users or designers. The two knowledge-based tools are seen as a whole system able to help designers make the best decisions. Therefore, the users cannot ask for a particular treatment.

The coupling process is iterative and is composed of two phases which correspond to the availability of knowledge:

- Filtering phase: the general knowledge-based system is used for filtering the constraints and for giving values to the *CSV* variables.
- Retrieval phase: the contextual knowledge-based system is used for retrieving relevant past cases and for advising the user with values to the *CBV* variables.

These two phases are combined in order to make the most of general and contextual knowledge. Two types of assistance have been identified:

- in the first one, the knowledge-based system is used for giving a rough idea of the values of the design variables: general and contextual knowledge are juxtaposed in order to complete the design. This first type is described in detail in Section 4.1,
- in the second type, the knowledge-based system is used for giving a more precise idea of the values of the design variables. In that case, general and contextual knowledge are interlaced in order to complete the design. This second type is described in detail in Section 4.2.

The proposed process is illustrated on a sub-set of variables of our industrial problem, presented in Section 4.3. It should be recalled, as stated in Section 1.3, that the union of the variables of the *CSP* part $CSV=\{Vcs\_k\}$ and of the variables of the *CBR* part $CBV=\{Vcb\_j\}$ corresponds to the design problem variables $DPV = CSV \cup CBV$ and that the intersection of these two subsets of design variables is nonempty: $CCV = CSV \cap CBV \neq \emptyset$.

We need to define a "fixed" variables set, noted *FVS*, which corresponds to the *DPV* variables valuated either by the user's inputs or by the filtering phase: these variables are non-negotiable, meaning that their value cannot be changed by the user. This set is used for the filtering and for the retrieval phases and can gather *CSV* and *CBV* variables.

4.1. Interactive knowledge juxtaposition

The two kinds of knowledge can be juxtaposed in order to give an idea of the current design. The two phases are carried out strictly in sequence: firstly, the filtering phase is done and restricts the domains of the *CSV* variables and secondly, the retrieval phase is launched on the *FVS* variables belonging to *CBV* in order to advise the user on the possible and realistic domain of the un-valuated *CBV* variables.

In this situation, the designer is going to describe her/his problem onto the design variables *DPV* step by step. After each user input, the knowledge-based system is going to use in sequence:

- Filtering phase on the *CSP* linking the *CSV* variables: the constraints *C_i* are then used to reduce the domains of the *CSV* variables.
- Retrieval phase on the *CBR* linking the *CBV* variables: it should be noted that as some of the *CBV* variables belong to *CCV* set, they can be valuated by the *CSP* filtering process. The retrieval process is then launched in order to complete the values of the un-valuated *CBV* variables thanks to the *CBV* variables belonging to *FVS* which already have a "fixed" value. In order to give advice about the solutions, an expert should have previously expressed the way of fulfilling each of the *CBV* variables, either by giving all the values, the maximum, the minimum, the interval of values, or the average value, on a maximal number *mn* of similar cases with a minimal similarity score *ms*.

After these two steps (filtering and retrieval), the domain of some *DPV* variables could have been modified: the designer can continue his/her design with a better knowledge of the possible solutions. Algorithm 1 summarizes this process.

Algorithm 1. KNOWLEDGE_JUXTA(*DPV*)

– ∴ – ***This algorithm is able to juxtapose general and contextual knowledge thanks to the design problem variables*** *DPV*.
– ∴ –*LoRV is the list of modified variables belonging only to CSV*.
– ∴ –*LoFV is the list of fixed variables belonging only to CBV*.
Begin
LoRV ← all the design variables of *CSV* with a modified domain
– ∴ – ***The algorithm starts by filtering all the*** *CSV* ***variables***.
While (*LoRV* ≠ ∅) **Do**
– ∴ – *v corresponds to the first variable of LoRV*
v ← POP(*LoRV*)
– ∴ – *The* CSP *model is filtered with the new domain of v*
FILTER_CSP(v)
LoRV ← all the new reduced variables of *CSV*
End While
– ∴ – *At this stage, all the variables of CSV have been reduced if necessary*
LoFV ← all the "fixed" variables of CBV : CBV ∩ FVS
– ∴ – ***The algorithm continues by giving values to the CBV variables***
– ∴ – *The search is made using the CBV ∩ FVS that have a non-negotiable value*.
SEARCH_CBR(*LoFV*, *mn*, *ms*)
– ∴ – *At this stage, all the variables of DPV have been reduced or advised if necessary*
Return *DPV*

4.2. Interactive knowledge interlacing

The two kinds of knowledge can be interlaced in order to make the most of contextual and general knowledge. In this situation, some variables belonging to *CSV* can be linked to some *CBV* variables by specific constraint.

We call these particular constraints, contextual constraints. They belong to the *CSP* model and have been previously identified and defined by an expert. Each of the contextual constraints is attached to a *CSV* variable and is described as follows:

- the list of *CBV* variables which are used in the constraint,
- the way of linking the *CBV* variables to the *CSV* variables,
- the maximal number mn_c of cases to use,
- the minimal score ms_c of similarity of the use cases.

Let us consider a variable $\alpha \subseteq CSV$. This variable is computed using the ratio of two variables β and $\lambda \subseteq CBV$. The contextual constraint *cc* expressing this particular knowledge is then described as follow: $cc(\alpha) = CBR\_request((\beta,\lambda), \alpha = \beta/\lambda, mn_c = 10, ms_c > 0.85)$.

In this case, the situation is quite the same as in the previous section: the designer is going to describe her/his problem onto the complete set of design variables *DPV* step by step. After each user's input, the knowledge-based system is going to use in sequence:

- Filtering phase, on the *CSP* linking the *CSV* variables. This phase is now decomposed into two sub-phases depending on the type of the constraints:
  - If a *CSV* variable v is constrained by a contextual constraint, then a *CBR* retrieval process is launched on the "fixed" variables $FSV \subseteq CBV$: the relevant cases are used to compute the reduced domain of v respecting the contextual constraint definition and the variables fulfilling definition,
  - else the constraints *C_i* are used to reduce the domains of the *CSV* variables.
- Retrieval phase on the *CBR* linking the *CBV* variables. The retrieval process is then launched another time, and follows exactly the same process described for previous knowledge juxtaposition. Here also the idea is to give advice to the user.

Table 1
Constraint C_1 between *MRB_JOB_HT*, *MRB_OPR_Struct-Mecha_LV* and *MRB_OPR_-Struct-Mecha_TR*.

C_1		
MRB_JOB_HT	*MRB_OPR_Struct-Mecha_LV*	*MRB_OPR_Struct-Mecha_TR*
Tiger	Complex	*tr_1*
Tiger	Light	*tr_5*
Puma	Complex	*tr_1*
Puma	Light	*tr_2*

Table 2
Constraint C_2 between *MRB_JOB_ST*, *MRB_OPR_Struct-Mecha_TR* and *OPR_Struct-Mecha_WLD*.

C_2		
MRB_JOB_ST	*MRB_OPR_Struct-Mecha_TR*	*MRB_OPR_Struct-Mecha_WLD*
Mini	{*tr_5*, *tr_2* }	[1, 500]
Interim	{*tr_5*, *tr_2*, *tr_1* }	[400, 1000]
Complete	{ *tr_1* }	[1000, 3000]

Table 3
Distance D_2 for the *EFF_OPR_Struct-Mecha_TR* variable.

D_2			
EFF_OPR_Struct-Mecha_TR	*tr_1*	*tr_2*	*tr_5*
tr_1	1	0.7	0.3
tr_2	0.7	1	0.5
tr_5	0.3	0.5	1

Table 4
Distance D_1 for the *EFF_OPR_Struct-Mecha_LV* variable.

D_1		
EFF_OPR_Struct-Mecha_LV	Complex	Light
Complex	1	0.3
Light	0.3	1

Algorithm 2 summarises this process.

Algorithm 2. KNOWLEDGE_INTER(*DPV*)

```
– ∴ – This algorithm is able to interlace general and contextual knowledge thanks to contextual constraints.
– ∴ – LoRV is the list of modified variables belonging to CSV.
– ∴ – LoFV is the list of fixed variables belonging to CBV.
Begin
  LoRV ← all the design variables of CSV with a modified domain
  – ∴ – The algorithm starts by filtering all CSV variables.
  While (LoRV ≠ ∅) Do
    – ∴ – v corresponds to the first variable of LoRV
    v ← POP(LoRV)
    If (v is associated to a contextual constraint) Then
      – ∴ – A CBR retrieval is launched for determining the contextual constraint.
      CBR_REQUEST((context_CBV_set),contextc_v,mn_c,ms_c)
    End If – ∴ – The CSP model is filtered taking into account the new domain of v
    FILTER_CSP(v)
    LoRV ← all the new reduced variables of CSV
  End While
  – ∴ – At this stage, all the variables of CSV have been reduced if necessary
  LoFV ← all the new "fixed" variables of CBV : CBV ∩ FVS
  – ∴ – The algorithm continues by giving a value to the design variables of CBV.
  – ∴ – The search is made using the CBV ∩ FVS that have a value given either by the user or deduct from the CSP filtering.
  SEARCH_CBR(LoFV,mn,ms)
  – ∴ – At this stage, all the variables of DPV have been reduced if necessary
  Return DPV
```

4.3. Application on the industrial problem

We illustrate the previous proposals on a sub-set of variables of our industrial problem, described in Section 2. Firstly, we define the simplified knowledge model and secondly, we propose two running scenarios corresponding to knowledge juxtaposition and interlacing processes.

4.3.1. Considered knowledge model

In this section, we describe the subset of design parameter variables *DPV* that are necessary to illustrate our proposals. We first present the variables that belong to both *CBR* and *CSP* models $\subseteq CCV$, then, the ones belonging only to the *CSP* model $\subseteq CSV$, then the ones belonging only to the *CBR* model $\subseteq CBV$ and we finish by the complete illustrative model.

CCV: *Constraint case variables*: As said previously, in Section 2.3, the job descriptions *MRB_JOB_V* belong to both models. In order to simplify our problem, we only consider two variables:

- helicopter type: *MRB_JOB_HT* with the domain {*Tiger*, *Puma*},
- maintenance job type: *MRB_JOB_ST* with the domain {*Complete*, *Interim*, *Mini*}.

CSV: *Constraint satisfaction variables*: As said previously, in Section 2.1, the maintenance operation descriptions *MRB*_OPR_*V* belong only to the *CSP* models. In order to simplify our problem, we only consider:

- one operation, the operation on the mechanical structure of the helicopter: *MRB_OPR_Struct-Mecha*, with two characteristics:
  - operation resource: *MRB_OPR_Struct-Mecha_TR* with the domain {*tr*_1, *tr*_2, *tr*_5},
  - operation level: *MRB_OPR_Struct-Mecha_LV* with the domain {*complex*, *light*},
- the cost indicator: *OPR_Struct-Mecha_CST* with the domain [1, 6000],
- the workload indicator: *OPR_Struct-Mecha_WLD* with the domain [1, 3000].

These variables are linked by two constraints:

- *c*1 links *MRB_JOB_HT*, *MRB_OPR_Struct-Mecha_LV* and *MRB_OPR_Struct-Mecha_TR* as described in Table 1,
- *c*2 links *MRB_JOB_ST*, *MRB_OPR_Struct-Mecha_TR* and *OPR_Struct-Mecha_WLD* as described in Table 2,

We notice that the variable *OPR_Struct-Mecha_CST* is not linked to the other variables because, the corresponding knowledge has not been modelled and is thus missing in the model.

CBV: *Case base variables*: As said previously, in Section 2.2, the effective maintenance operation descriptions *EFF_OPR_V* belong only to the *CBR* models. In order to simplify our problem, we only consider:

- one operation, the operation on the mechanical structure of the helicopter: *EFF_OPR_Struct-Mecha*, with two characteristics:
  - operation resource: *EFF_OPR_Struct-Mecha_TR* with the distances d_2 described in Table 3,
  - operation level: *EFF_OPR_Struct-Mecha_LV* with the distances d_1 described in Table 4,
- the cost indicator: *OPR_Struct-Mecha_CST_EFF*,
- the workload indicator: *OPR_Struct-Mecha_WLD_EFF*.

We also need to identify the distances between the possible values of *MRB_JOB_HT* and *MRB_JOB_ST*. These distances are d_3

described in Table 5 for the helicopter type and d_4 described in Table 6 for the service type.

For these *CBV* variables, when many cases need to be considered for treatment (advice and contextual constraint filtering process), an expert has identified the way of fulfilling them from past cases:

- *EFF_OPR_Struct-Mecha_TR*: all the values of the operation resources are given,
- *EFF_OPR_Struct-Mecha_LV*: all the values of the operation levels are given,
- *OPR_Struct-Mecha_CST_EFF*: the interval of relevant cost values is retrieved,
- *OPR_Struct-Mecha_WLD_EFF*: the average workload value is retrieved,
- *MRB_JOB_HT*: all the helicopter types values are given,
- *MRB_JOB_ST*: all the service types values are given.

Table 5
Distance D_3 for the *MRB_JOB_HT* variable.

D_3		
MRB_JOB_HT	Puma	Tiger
Puma	1	0.8
Tiger	0.8	1

Table 6
Distance D_4 for the *MRB_JOB_ST* variable.

D_4			
MRB_JOB_ST	Mini	Interim	Complete
Mini	1	0.8	0.1
Interim	0.8	1	0.6
Complete	0.1	0.6	1

The similarity function *F* of the cases is computed as the mean between the four previous distances: $F=(d_1+d_2+2*d_3+d_4)/5$. For computing the similarity score of a case, we consider the distance between values of the fixed variables *FVS* and the value 1 for the others. The case base contains five past cases as presented in Table 7.

Complete model: The complete illustrative model is then composed of 10 design problem variables with two variables $\subseteq CCV$, six variables $\subseteq CSV$ and six variables $\subseteq CBV$, as shown if Fig. 12.

4.3.2. Running scenario

In this paragraph, we illustrate the juxtaposition and interlacing processes presented respectively in Sections 4.1 and 4.2. Firstly, we consider only the interactive knowledge juxtaposition without any contextual constraints. Secondly, we add to the *CSP* model a contextual constraint in order to compute the cost of the operation *OPR_Struct-Mecha_CST*.

At the beginning, the user has the following information on the *DPV*:

- on the *CCV*, without filtering or retrieval process, all the values are possible and compliant with the *CSP* and the *CBR* models:
  - *MRB_JOB_HT*={*Tiger*, *Puma*},
  - *MRB_JOB_ST* = {*Complete*, *Interim*, *Mini*},
- on the *CSV*, without any filtering, all the values are possible:
  - *MRB_OPR_Struct-Mecha_TR*={*tr*_1, *tr*_2, *tr*_5},
  - *MRB_OPR_Struct-Mecha_LV*={*complex*, *light*},

Table 7
CBR database.

MRB_JOB_HT	*MRB_JOB_ST*	*EFF_OPR_Struct-Mecha_LV*	*EFF_OPR_Struct-Mecha_TR*	*OPR_Struct-Mecha_CST_EFF*	*OPR_Struct-Mecha_WLD_EFF*
Tiger	Mini	Light	*tr*_5	650	550
Tiger	Mini	Light	*tr*_5	600	300
Tiger	Mini	Light	*tr*_5	630	400
Tiger	Interim	Light	*tr*_1	1500	850
Tiger	Interim	Light	*tr*_1	1300	750

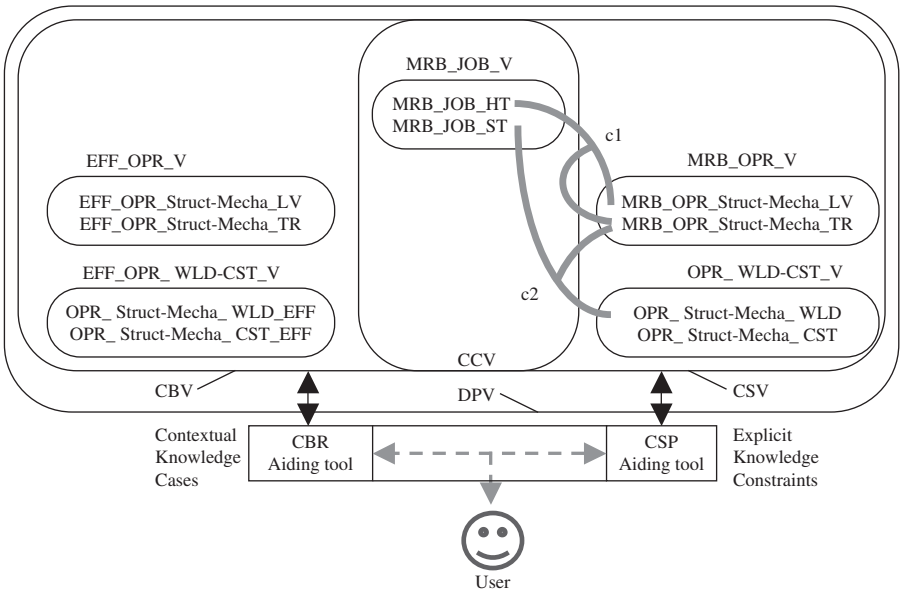

Fig. 12. Knowledge juxtaposition model of the example.

- *OPR_Struct-Mecha_CST*=[1, 6000],
- *OPR_Struct-Mecha_WLD*=[1, 3000],

- on the *CBV*, all the cases give their values as advised values:
  - *EFF_OPR_Struct-Mecha_TR*={ *tr*_5, *tr*_1} because all the values are given as advice,
  - *EFF_OPR_Struct-Mecha_LV*={*light*} because all the values are given as advice,
  - *OPR_Struct-Mecha_CST_EFF*={[600, 1500]} because the overall interval of values is retrieved,
  - *OPR_Struct-Mecha_WLD_EFF*=570 because the average value is retrieved.

For both of the running examples, we consider only three user inputs:

- *MRB_JOB_HT*=*Tiger*,
- *MRB_JOB_ST*=*Complete*,
- *OPR_Struct-Mecha_WLD*=2500, at the end, before storing the case.

The filtering algorithms are based on arc-consistency as defined by Mackworth and Freuder (1985) for discrete variables and Lhomme (1993) for continuous ones.

At the end of the maintenance service, the user can complete the effective parameters and ask to store the case.

Interactive knowledge juxtaposition: After the first user's input *MRB_JOB_HT*=*Tiger*, the knowledge-based system is going to use in sequence:

- Filtering phase on the *CSV* variables. The filtering mechanism deduces:
  - unchanged: *MRB_JOB_ST*={*Complete*, *Interim*, *Mini*},
  - changed: *MRB_OPR_Struct-Mecha_TR*={*tr*_1, *tr*_5} via the constraint c_1,
  - unchanged: *MRB_OPR_Struct-Mecha_LV*={*complex*, *light*},
  - unchanged: *OPR_Struct-Mecha_CST*=[1, 6000],
  - unchanged: *OPR_Struct-Mecha_WLD*=[1, 3000].
- Retrieval phase on the *CBV* variables. The retrieving mechanism advises via the request SEARCH_CBR(*LoFV*={*MRB_JOB_HT*=*Tiger*}, *mn* = 5, ms > 0.9) the following domains. The similarity score for each case is presented in Table 8:
  - *EFF_OPR_Struct-Mecha_TR*={ *tr*_5, *tr*_1},
  - *EFF_OPR_Struct-Mecha_LV*={*light*},
  - *OPR_Struct-Mecha_CST_EFF*={[600, 1500]},
  - *OPR_Struct-Mecha_WLD_EFF*=570.

After the second user's input *MRB_JOB_ST*=*Complete*, the knowledge-based system is going to use in sequence:

- Filtering phase on the *CSV* variables. The filtering mechanism deduces:
  - changed: *MRB_OPR_Struct-Mecha_TR*={*tr*_1} via the constraint c_2,
  - changed: *MRB_OPR_Struct-Mecha_LV*={*complex*},via the constraints c_2 and c_1,
  - unchanged: *OPR_Struct-Mecha_CST*=[1, 6000],
  - changed: *OPR_Struct-Mecha_WLD*=[1000, 3000] via the constraint c_2.
- Retrieval phase on the *CBV* variables. The retrieving mechanism advises via the request SEARCH_CBR(*LoFV*={*MRB_JOB_HT*=*Tiger*, *MRB_JOB_ST*=*Complete*}, *mn*=5, *ms* > 0.9) the following domains from cases 4 and 5. The similarity score for each case is presented in Table 9:
  - *EFF_OPR_Struct-Mecha_TR*={*tr*_1},
  - *EFF_OPR_Struct-Mecha_LV*={*light*},
  - *OPR_Struct-Mecha_CST_EFF*={[1300, 1500]},
  - *OPR_Struct-Mecha_WLD_EFF*=800.

The only difference lies in the advice relevant to variable *EFF_OPR_Struct-Mecha_TR* which proposes *tr*_1 instead of *tr*_1 and *tr*_5.

At the end of the service, the user can complete the effective parameters and ask to store the case. For instance, (s)he can arrive at the following result:

- user's inputs:
  - *MRB_JOB_HT*={*Tiger*},
  - *MRB_JOB_ST*={*Complete*},
- *CSP* deduced values:
  - *MRB_OPR_Struct-Mecha_TR*={*tr*_1},
  - *MRB_OPR_Struct-Mecha_LV*={*light*},
  - *OPR_Struct-Mecha_CST*=[1, 6000]. The user does not know the estimated cost, so (s)he can leave the complete interval,
  - *OPR_Struct-Mecha_WLD*=2500, last user's input, consistent with the previous domain [1000, 3000],
- *CBR* effective values:
  - *EFF_OPR_Struct-Mecha_TR*={*tr*_5}. In the reality, the *tr*_1 resource was not free, so (s)he has used the *tr*_5 one instead,

Table 8
Cases similarity score for the first user's input.

MRB_JOB_HT	*MRB_JOB_ST*	*EFF_OPR_Struct-Mecha_LV*	*EFF_OPR_Struct-Mecha_TR*	*OPR_Struct-Mecha_CST_EFF*	*OPR_Struct-Mecha_WLD_EFF*	*F*
Tiger	Mini	Light	*tr*_5	650	550	1
Tiger	Mini	Light	*tr*_5	600	300	1
Tiger	Mini	Light	*tr*_5	630	400	1
Tiger	Interim	Light	*tr*_1	1500	850	1
Tiger	Interim	Light	*tr*_1	1300	750	1

Table 9
Cases similarity score for the second user's input.

MRB_JOB_HT	*MRB_JOB_ST*	*EFF_OPR_Struct-Mecha_LV*	*EFF_OPR_Struct-Mecha_TR*	*OPR_Struct-Mecha_CST_EFF*	*OPR_Struct-Mecha_WLD_EFF*	*F*
Tiger	Mini	Light	*tr*_5	650	550	0.82
Tiger	Mini	Light	*tr*_5	600	300	0.82
Tiger	Mini	Light	*tr*_5	630	400	0.82
Tiger	Interim	Light	*tr*_1	1500	850	0.92
Tiger	Interim	Light	*tr*_1	1300	750	0.92

- *EFF_OPR_Struct-Mecha_LV*={*light*},
- *OPR_Struct-Mecha_CST_EFF*={760}, corresponding to the effective cost,
- *OPR_Struct-Mecha_WLD_EFF*={420}, corresponding to the effective workload.

If (s)he wants, (s)he can ask to store the values of the *CBV* into the *CBR* database.

Interactive knowledge interlacing: Now, we add to the *CSP* model a numerical constraint nc_1 and a contextual constraint cc_1 in order to compute the cost of the operation *OPR_Struct-Mecha_CST*. The numerical constraint nc_1 links *OPR_Struct-Mecha_CST* and *OPR_Struct-Mecha_WLD* thanks to the variable α with the domain $D_\alpha = [1,2]$: $OPR\_Struct\text{-}Mecha\_CST = \alpha * OPR\_Struct\text{-}Mecha\_WLD$. The contextual constraint cc_1 links α to the ratio of two *CBV* variables (*OPR_Struct-Mecha_CST_EFF*, *OPR_Struct-Mecha_WLD_EFF*). This constraint cc_1 is then described as follows: $cc_1(\alpha) = CBR\_request((OPR\_Struct\text{-}Mecha\_CST\_EFF, OPR\_Struct\text{-}Mecha\_WLD\_EFF), \alpha = OPR\_Struct\text{-}Mecha\_CST\_EFF / OPR\_Struct\text{-}Mecha\_WLD\_EFF, mn_c = 10, ms_c > 0.85)$.

The updated illustrative model is then composed of 10 design problem variables with two variables $\subseteq CCV$, six variables $\subseteq CSV$, six variables $\subseteq CBV$, and one computational variable as shown in Fig. 13.

After the first user's input *MRB_JOB_HT*=*Tiger*, the knowledge-based system is going to use in sequence:

- Filtering phase on the *CSV* variables. The filtering mechanism deduces:
  - unchanged: *MRB_JOB_ST*={*Complete*, *Interim*, *Mini*},
  - changed: *MRB_OPR_Struct-Mecha_TR*={*tr*_1, *tr*_5} via the constraint c_1,
  - unchanged: *MRB_OPR_Struct-Mecha_LV*={*complex*, *light*},
  - unchanged: $\alpha = [1,2]$. There is no need to filter this variable because neither *OPR_Struct-Mecha_CST* nor *OPR_Struct-Mecha_WLD* have been modified,
  - unchanged: *OPR_Struct-Mecha_CST*=[1, 6000],
  - unchanged: *OPR_Struct-Mecha_WLD*=[1, 3000],
- Retrieval phase on the *CBV* variables with exactly the same results and process as juxtaposition of knowledge. The retrieving mechanism advises via the request SEARCH_CBR(*LoFV*=*MRB_JOB_HT*=*Tiger*}, mn=5, $ms > 0.9$) the following domains. The similarity score for each case is presented in Table 8:
  - *EFF_OPR_Struct-Mecha_TR*={ *tr*_5, *tr*_1},
  - *EFF_OPR_Struct-Mecha_LV*={*light*},
  - *OPR_Struct-Mecha_CST_EFF*={[600, 1500]},
  - *OPR_Struct-Mecha_WLD_EFF*=570.

After the second user's input *MRB_JOB_ST*=*Complete*, the knowledge-based system is going to use in sequence:

- Filtering phase on the *CSV* variables. The filtering mechanism deduces:
  - changed: *MRB_OPR_Struct-Mecha_TR*={*tr*_1} via the constraint c_2,
  - changed: *MRB_OPR_Struct-Mecha_LV*={*complex*},via the constraints c_2 and c_1,
  - changed: $\alpha = [1.625, 1.875] = [1300, 1500] \oslash 800$, via the contextual constraints cc_1 (the two last cases of Table 9 are retrieved),
  - changed: *OPR_Struct-Mecha_CST*={[1625, 5625]} via the constraints nc_1:

$$OPR\_Struct\text{-}Mecha\_CST = \alpha * OPR\_Struct\text{-}Mecha\_WLD \Leftrightarrow D_{OPR\_Struct\text{-}Mecha\_CST} = \{D_\alpha * D_{OPR\_Struct\text{-}Mecha\_WLD}\} \cap D_{OPR\_Struct\text{-}Mecha\_CST} = \{[1.625, 1.875] \otimes [1000, 3000]\} \cap [1, 6000] = \{[1625, 5625]\} \cap [1, 6000]$$

  - changed: *OPR_Struct-Mecha_WLD*=[1000, 3000], via the constraint c_2.

We can see that considering a contextual constraint allows us to have more information about some variables. This information is always up-to-date thanks to the most relevant past cases. We can then estimate better the values of some

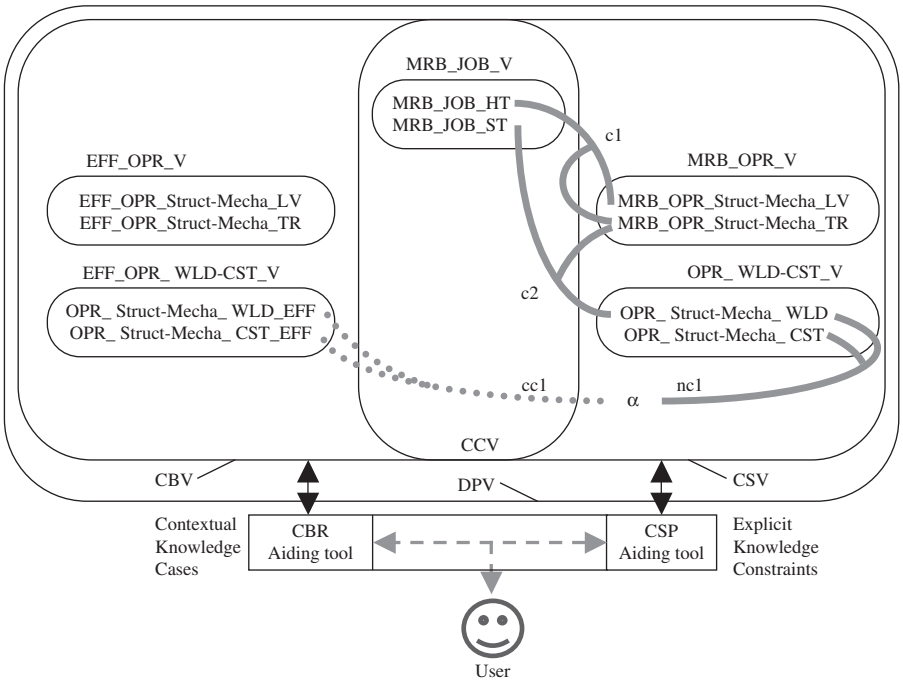

Fig. 13. Knowledge interlacing model of the example.

variables, in our case the theoretical cost from [1, 6000] without any contextual knowledge to [1625, 5625] considering contextual knowledge and user inputs.
- Retrieval phase on the *CBV* variables. The retrieving mechanism advises via the request SEARCH_CBR(*LoFV* = {*MRB_JOB_HT*=*Tiger*, *MRB_JOB_ST*=*Complete*}, *mn*=5, *ms* > 0.9) the following domains. The similarity score for each case is presented in Table 9:
  - *EFF_OPR_Struct-Mecha_TR*={*tr*_1},
  - *EFF_OPR_Struct-Mecha_LV*={*light*},
  - *OPR_Struct-Mecha_CST_EFF*={[1300, 1500]}, only cases 4 and 5 are used,
  - *OPR_Struct-Mecha_WLD_EFF*=800, only cases 4 and 5 are used.

At the end of the service, the user can complete the effective parameters and ask to store the case.

- user's inputs:
  - *MRB_JOB_HT*={*Tiger*},
  - *MRB_JOB_ST*={*Complete*},
- *CSP*deduced values:
  - *MRB_OPR_Struct-Mecha_TR*={*tr*_1},
  - *MRB_OPR_Struct-Mecha_LV*={*light*},
  - *OPR_Struct-Mecha_CST*=[1333.35, 1538.46], values deduced from the contextual constraint and the value of *OPR_Struct-Mecha_WLD*=2500. The user can estimate the cost to 1500, for instance,
  - *OPR_Struct-Mecha_WLD*=2500, last user input, consistent with the previous domain [1000, 3000],
- *CBR* effective values:
  - *EFF_OPR_Struct-Mecha_TR*={*tr*_5}. In reality, the *tr*_1 resource was not free, so (s)he has used the *tr*_5 one instead,
  - *EFF_OPR_Struct-Mecha_LV*={*light*},
  - *OPR_Struct-Mecha_CST_EFF*={760}, corresponding to the effective cost,
  - *OPR_Struct-Mecha_WLD_EFF*={420}, corresponding to the effective workload.

If (s)he wants, (s)he can ask to store the values of the *CBV* into the *CBR* database.

4.3.3. Synthesis

In this section, we have proposed a complete coupling process using at the same time general and contextual knowledge in order to help designers make the best decisions.

This process is iterative and is able to juxtapose and interlace knowledge deduced from the *CSP* and the *CBR* depending on the availability of knowledge. For each of the assistance types, we have proposed an algorithm that explains the aiding design process and we have illustrated it on a simplified industrial problem.

Our approach is quite original because firstly, the two tools really work together and not in a sequential way, as it has been previously studied: they have to exchange and share knowledge in order to find a solution. Secondly, they are used in an interactive aiding design process, meaning that designers input progressively their design requirements in order to complete the current design. Our approach has been developed and tested on the industrial example presented in Section 2.

5. Conclusion

In this paper, we have presented a new way of associating general and contextual knowledge in order to help designers make the best decisions.

We have begun with a reminder of the different ways of modelling these two kinds of knowledge and by using *Artificial Intelligence* tools as *CSP* and *CBR*. We have then reviewed the process of each of these tools and explained how knowledge are modelled in such tools. We have also presented the industrial problem which is at the origin of our proposals.

We have then reviewed on the studies which have already mixed *CSP* and *CBR* approaches. These studies have focused on the validation, on the enrichment and on the sequential uses of the two kinds of knowledge. The sequential uses do not combine the two kinds of knowledge on a same design decision as the designer might expect.

We have therefore proposed a complete coupling process using general and contextual knowledge at the same time in order to help designers make better decisions. This process corresponds to the use of the two tools in a simultaneous way in order to make the most of general and contextual knowledge. Two types of assistance have been identified depending on the availability of knowledge:

- In the first one, the knowledge-based system juxtaposes the general and contextual knowledge. General knowledge prunes the solution space while contextual knowledge is used to provide advices to the user,
- The second one, the knowledge-based system interlaces the two types of knowledge in order to give more accurate information to the designer. This information corresponds to some specific constraints, named contextual constraints . The contextual constraints link variables belonging to the *CSP* and to the *CBR* models and always use updated knowledge thanks to the past cases database.

Our knowledge system matches designers' expectations: the two types of knowledge are complementary and their simultaneous use provides more accurate results and better quality information. Their mix (juxtaposition or interlacing) depend on the capability and the skill of the experts to express design relations (constraints, contextual constraints, distance similarity, value fulfillment). The designers can then benefit from general and contextual information in a single system. Our approaches have been developed and tested on an industrial example that highlights the kind of assistance that our knowledge system can provide.

Concerning the contextual constraints, it could be very interesting to be able to select some of them or to tune them depending on some *CSP* variables, to link several *CSV* to several *CBV* in a single constraint and to search on cases with several *CBV* values. We can also use data mining to extract rules from the relevant cases supplied by the *CBR*. These rules can be translated into constraints and inputted in the *CSP* model during the filtering process, under specific conditions.

Acknowledgements

The authors wish to acknowledge the *Aerospace Valley* Association and all partners in the project, and in particular the experts of *Helimaintenance R & D* for their involvement in the building of the illustrative model.

References

Aamodt, A., Plaza, E., 1994. Case-based reasoning: foundational issues, methodological variations, and system approaches. AICom—Artif. Intell. Commun. 7 (1), 39–59.

Althoff, K.D., 2008. Advances in Case-based Reasoning: 9th European Conference. ECCBR 2008, Trier, Germany, September 1–4, Springer. ISBN 3540855017.

Bergmann, R., Althoff, K.-D., Breen, S., Göker, M., Manago, M., Traphöner, R., Wess, S., 2003. Developing Industrial Case-Based Reasoning Applications. Lecture Notes in Artificial Intelligence, vol. 1612Springer Verlag.

Bichindaritz, I., Marling, C., 2006. Case-based reasoning in the health sciences: What's next? Artif. Intell. Med. 36 (2), 127–135 ISSN 0933-3657.

Bin, E., Emek, R., Shurek, G., Ziv, A., 2010. Using a constraint satisfaction formulation and solution techniques for random test program generation. IBM Syst. J. 41 (3), 386–402 ISSN 0018-8670.

Bodirsky, M., Dalmau, V., 2006. Datalog and constraints satisfaction with infinite templates. STACS, vol. 3884. , Springer, pp. 646–659.

Brown, D.C., Chandrasekaran, B., 1984. Expert systems for a class of mechanical design activity in knowledge engineering. In: Proceedings of the IFIP WG 5.2 Working Conference on Knowledge Engineering in Computer-Aided Design, Budapest, Hungary, pp. 259, 17–19.

Carsten, T., 2001. Customizing Software Engineering Experience Management Systems to Organizational Needs. Ph.D. Thesis in Experimental Software Engineering, Band 4, Fraunhofer IRB Verlag, ISBN 978-3-8167-5881-5.

Changchiena, S.W., Linb, M.-C., 2005. Design and implementation of a case-based reasoning system for marketing plans. Expert Syst. Appl. 28 (1), 43–53 ISSN 0957-4174.

Chenouard, R., Granvilliers, L., Sebastian, P., 2009. Search heuristics for constraint-aided embodiment design. In: AIEDAM, vol. 23(2), pp. 175–195.

Coyne, R.D., Rosenman, M.A., Radford, A.D., Balachandran, M., Gero, J.S., 1989. Knowledge-Based Design SystemsAddison-Wesley Longman Publishing Co., Inc., Boston, MA, USA ISBN 0201103818.

Dutta, S., Wierenga, B., Dalebout, A., 1997. Designing management support systems using an integrative perspective. Commun. ACM 40 (6), 70–79 ISSN 0001-0782.

Fargier, H., Lang, J., Schiex, T., 1996. Mixed constraint satisfaction: a framework for decision problems under incomplete knowledge. In: Proceedings of AAAI-96, pp. 175–180.

Felfernig, A., Friedrich, G., Isak, K., Shchekotykhin, K., Teppan, E., Jannach, D., 2007. Automated debugging of recommender user interface descriptions. Appl. Intell. 31 (1), 1–14 ISSN 0924-669X. doi: 10.1007/s10489-007-0105-8.

Goel, A.K., Craw, S., 2006. Design, innovation and case-based reasoning. Knowl. Eng. Rev. 20 (3), 271–276. ISSN: 0269-8889, 〈http://hdl.handle.net/10059/55〉, doi: 10.1017/S0269888906000609.

Inakoshi, H., Okamoto, S., Ohta, Y., Yugami, N., 2001. In: Effective Decision Support for Product Configuration by Using CBR. Fujitsu Labs, Japan.

Kolodner, J.L., 1993. Rag Case-Based ReasoningMorgan Kaufmann Publishers 696 pp. ISBN 978-1558602373.

Lhomme, O., 1993. Consistency techniques for numeric CSP. In: International Joint Conference on Artificial Intelligence, vol. 13, pp. 232–238.

Lopez, B., 2003. Holiday scheduling for city visitors. In: Frew, A., et al. (Eds.), Information and Communication Technologies in Tourism, Springer Computer Science.

Mackworth, A.K., Freuder, E., 1985. The complexity of some polynomial network-consistency algorithms for Constraint Satisfaction Problems. Artif. Intell. 25 (1), 65–74. doi:10.1016/0004-3702(85)90041-4 ISSN 0004-3702.

Minsky, M., 1974. A Framework for Representing Knowledge. AI Memos.

Nemati, H.R., Steiger, D.M., Iyer, L.S., Herschel, R.T., 2002. An architectural integration of knowledge management, decision support, artificial intelligence and data warehousing. Decision Support Syst. 33 (2), 143–161 ISSN 0167-9236, doi: 10.1016/S0167-9236(01)00141-5.

Norme NF X60012, 2006. Maintenance. Termes et définitions des éléments constitutifs des biens et de leur approvisionnement.

Poncelin, G., Glade, M., Cauvin, A., Dufrene, D., Lyonnet, P., 2006. Design to maintenance cost by the control of the products environment, Product Life Cycle AspectsSpringer, pp. 95–107, doi: 10.1007/978-2-287-48370-7. ISBN 978-2-287-48363-9.

Purvis, L., Pu, P., 1995. Adaptation using constraint satisfaction techniques. In: Case-Based Reasoning Research and Development, Springer, pp. 289–300.

Riesbeck, E.-l., Shank, R., 1989. Inside Case-Based ReasoningLawrence Erlbaum Associates, New Jersey 452 pp. ISBN 978-0898597677.

Roldan, E., Negny, S., Le Lann, J.M., Cortes, G., 2010. Constraint satisfaction problem for case-based reasoning adaptation: application in process design. In: Pierucci, S., Buzzi Ferraris, G. (Eds.), 20th European Symposium on Computer Aided Process Engineering, ESCAPE20. Elsevier.

Ruet, M., Geneste, L., 2002. Search and adaptation in fuzzy object oriented case base. In: Advances in Case-Based Reasoning, Springer, pp. 723–730.

Sqalli, M.H., Freuder, E.C., 1998. Integration of CSP and CBR to compensate for incompleteness and incorrectness of models. In: The AAAI-98 Spring Symposium on Multimodal Reasoning, March 23–25, 1998. Stanford University, California, USA.

Sqalli, M.H., Purvis, L., Freuder, E.C., 1999. Survey of applications integrating constraint satisfaction and case-based reasoning. In: The First International Conference and Exhibition on the Practical Application of Constraint Technologies and Logic Programming (PACLP99), April 19–21, 1999, London, UK.

Vareilles, E., Aldanondo, M., Gaborit, P., 2007. Evaluation and design: a knowledge-based approach. Int. J. Comput. Integrated Manuf. 20 (7), 639–653 doi: 10.1080/09511920701566517.

White, J., Dougherty, B., Schmidt, D.C., Benavides, D., 2009. Automated reasoning for multi-step software product-line configuration problems. In: Proceedings of the Software Product Line Conference, pp. 11–20.