Case Combination for Conceptual Design

B.D. Netten[1], **R.A. Vingerhoeds**[1,2]

[1)] Delft University of Technology, Faculty of Technical Mathematics and Informatics, Julianalaan 132, 2628 BL Delft, Netherlands, [bart | rob]@kgs.twi.tudelft.nl

[2)] University of Wales Swansea, Dept. of Electrical and Electronic Engineering, Singleton Park, Swansea SA2 8PP, United Kingdom

EADOCS (expert assisted design of composite structures) is a multi-level conceptual design system, in which case-based reasoning is applied to quantify a qualitative prototype solution. Design specifications and targets for case-based retrieval comprise functional requirements, preferred and prototype solutions. Specified design problems cannot be decomposed into tractable subproblems for the retrieval and integration of subsolutions. Case adaptation requires extensive numerical design analysis, heuristic modification and optimisation. Combination of case parts is often necessary to solve multiple functional requirements more efficiently. Case merging can strongly improve the starting points and reduce design analysis for heuristic modification and numerical optimisation for case revision. In EADOCS, combination of interdependent components is possible under certain conditions on the solution. This paper addresses the approach, approximations and conditions to reuse interdependent case parts.

1 Conceptual Design Process

A conceptual design phase starts with the specification of design requirements and preferences. Specifications not only represent a design problem, they also express a designer's experience for practical solutions, feasibility and optimality. The specification of a design problem for retrieval comprises hard and soft requirements for design functionality and prototype design solutions. The final result should be the "best" conceptual design as the input for the preliminary design phase. Preliminary and detailed design phases are then supported by numerical design tools for which a good initial conceptual design is required as a starting point. Currently, these are commonly generated manually.

EADOCS is developed to support this conceptual design process by a multi-level expert system in which constraint-, case- and rule-based reasoning techniques are applied. EADOCS focusses on thin-walled fibre reinforced composite panels in aircraft structures. In this application domain, design functionality is strongly dependent on component interactions. This has three consequences for retrieval and reuse:

© Netten, Vingerhoeds, 1996

- Design subproblems cannot be causally related to specific components for which a case subsolution can be retrieved.
- Operations can only be defined for modification of component behaviour and not for the functionality of the complete design.
- Component behaviour for current specifications cannot be retrieved from the case-base, but requires extensive numerical analysis.

Composition of solutions is, in the current application domain, well defined from background knowledge. Designs are statically decomposed into components. A sandwich panel is decomposed into skin and core components, see Figure 1. The skin is further decomposed into layers. A layer is a package of standard plies of the same material and with same fibre orientation.

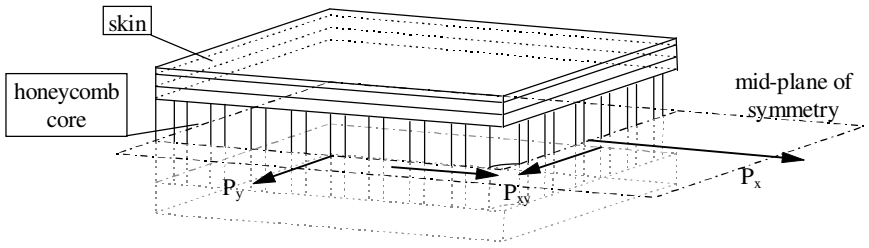

Figure 1: Symmetric sandwich panel

Designs are declared at two levels of abstraction:

Prototype design a qualitative design representation by solution classes.

Conceptual design a quantitative design representation by instantiation of objects for the prototype solution classes, with attribute values assigned (parametric description).

Qualitative design variables for the selection of a prototype are the solution classes. Design variables for conceptual selection and modification are defined by objects and attributes. Attributes are symbolic, discrete or continuous numerical variables.

Design problems are specified by functional requirements, preferred solutions and optimality criteria. A design should provide the functionality under different operational conditions, and requirements are specified for several combinations of functionality and operational conditions. Functionality is represented in a hierarchical class structure, in which the requirements are specified as objects.

Each class of solutions can be characterised by typical modes of behaviour. Components strongly affect other component behaviour. Evaluation of design behaviour and functionality is only possible by evaluation of interactive behaviour of all design components. Numerical analysis routines are available for evaluation of

© Netten, Vingerhoeds, 1996

component and design behaviour. These routines can only be applied for adaptation in the final phase when a conceptual design is completely defined. The numerical analysis and optimisation routines have two other disadvantages for application in conceptual design.

1. Many design parameters are symbolic or discrete, for which numerical optimisation routines cannot be efficiently applied.
2. Numerical optimisation requires considerable computational effort and solves only a part of the conceptual design problem.

The initial solution strongly affects the result of numerical optimisations. Heuristics and numerical optimisations are only applicable for local search in the non-linear design space, and are sensitive to local optima. The starting point should have optimal values for symbolic variables to optimise numerical values. Otherwise, computational efforts are wasted, because search is performed in less favourable design spaces.

Transformation of case solutions and subsolutions require accurate models to predict component and design behaviour for the specifications. Relations for interactive component behaviour are too complex to be modelled in specialist or strategic adaptation heuristics for the reuse of a retrieved design or component. Specifications cannot be decomposed into tractable subproblems that can be solved independently for a design or component. Approximations are defined for heuristic modification operations. These approximations are derived by inversion of behavioural relations for a single component and a particular mode of behaviour. These operations, however, do not take interaction of other components or specifications into account. Behaviour interaction has to be verified by numerical analysis in a revision step.

The highly non-linear and interdependent component behaviour poses two major restrictions to incremental retrieval of subsolutions for adaptation.

1. Specified design problems cannot be decomposed into subproblems for a component. Retrieval of a subsolution for a subproblem cannot be restricted to a class of components.
2. Retrieved solutions cannot be adapted in the reuse step to satisfy specified requirements, and retrieved subsolutions cannot be adapted in the reuse step for integration into a new solution.

The first restriction implies that it is not possible to formulate a subproblem for a specific class of case components only, and to retrieve and integrate a solution from this class. In other words, solutions to subproblems cannot be expressed as parts or snippets of previous design cases, and behaviour of components cannot be abstracted to predict overall design behaviour as in CADET [Sycara et al. ‘92]. The most common approach to retrieval of cases with similar problems are not possible for retrieving case parts for subproblems, as applied also in for example JULIA

© Netten, Vingerhoeds, 1996

[Hinrichs and Kolodner '91], Déjà Vu [Smyth and Keane '95], or PARIS [Bergmann and Wilke '96]. Dynamic problem decomposition to solve a subproblem at a lower level of abstraction, as for example in Déjà Vu and PARIS, is not possible either.

The second restriction implies that adaptation knowledge cannot be applied to guide retrieval for subproblems for design functionality, as in Déjà Vu. The approach presented in [Börner '93] requires transformation relations for modification and their inverse relations for behaviour, on the level of design requirements. In current application, these relations are only available in this form on detailed level of components.

The objective of this paper is to address the issue of case merging, related specifically to conceptual design, and in particular focussing on thin-walled composite reinforced structures.

2 Conceptual Design Phases

Design specifications, solutions and their behaviour are considered at two levels of abstraction for prototypes and concepts. These levels correspond to three design phases (Figure 2);

1. Prototype selection; qualitative design of prototype solutions.
2. Concept selection; quantification of a prototype solution.
3. Concept modification and optimisation; quantitative adaptation.

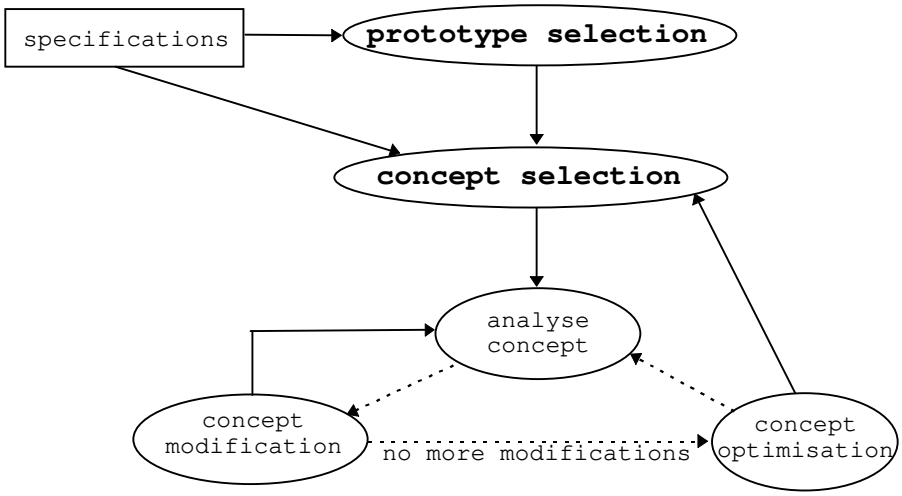

Figure 2: Conceptual design process

© Netten, Vingerhoeds, 1996

2.1 Prototype Selection

In the first design phase, the objective is to reduce the search space to feasible classes of solutions for the prototype design and components. Prototype solutions are assembled by selection of solution classes that can be composed and integrated into a feasible design. Optimality is estimated to select one or more alternative prototypes for following design phases. Prototype selection requires semi-qualitative reasoning about design problems and solutions. The selection process is formulated at the abstraction level of prototype designs:

- Functional requirements are categorised into typical problems with a qualitative interpretation. A functional category is defined by a pattern on attribute values from a functional class. For each functional class, attribute values are initially discretised into intervals of continuous or discrete numerical variables or into subsets of nominal values. Functional requirements are specified as typical categories or discretised onto these categories.
- Composition of prototype designs is determined by the solution class hierarchy. At each aggregation level one or more solutions can be selected from specialised classes for the components.
- Each solution class is characterised by a prototypical behaviour with respect to functionality, optimality criteria or integration with other solution classes. From theoretical and semi-empirical behavioural relations, design parameters can be determined to characterise prototypical behaviour. Design parameters are expressions from functional and solution attributes, and are also discretised as secondary features. Categories of behaviour can be defined for classes of solutions, similar to functional categories. Relative performance can be expressed as a qualitative constraint on prototypical behaviour for each solution class and category.

Constrained-based reasoning can be applied to assemble prototype designs from classes of solutions for the components. These solution classes are selected for their prototypical behaviour for required functionality, integration and optimality.

2.2 Concept Selection

The second design task is to quantify a selected prototype solution. Quantification requires the transformation of a qualitative solution into a quantitative solution that satisfies the specified requirements and optimality criteria. Due to the strong interdependence of component behaviour, such transformations cannot be declared from scratch by instantiation of solution objects. Quantification is divided in subtasks

© Netten, Vingerhoeds, 1996

for retrieval of similar cases, instantiation of case solutions and subsolutions, and composition of subsolutions into an initial conceptual design for revision.

The initial target for retrieval is declared by primary and secondary features for specified functionality and the selected prototype solution. Initially, a complete solution is retrieved as an initial conceptual design from a single case. Parts of the solutions of other cases can be retrieved and substituted under certain conditions. The objective for incremental retrieval is to reuse as much information from the case-base as possible, before performing computationally expensive revisions. Conflicts in behaviour for integration of subsolutions are not evaluated or solved during concept selection. In section 3, this incremental retrieval and reuse process will be described in more detail.

The similarity is measured by pattern matching on the objects and attributes in two steps. The value domains of each attribute are divided into intervals of continuous or discrete numerical variables, or into subsets of nominal values. The values for the loading conditions, for example, are divided as in **Error! Reference source not found.**. The combination of attribute subdomains represents a typical category of objects within a class. The first loading conditions is classified in the category of ($P_x \in < -\infty, -4000]$, $P_x \in < -500, 500]$, $P_{xy} \in < -500, 500]$).

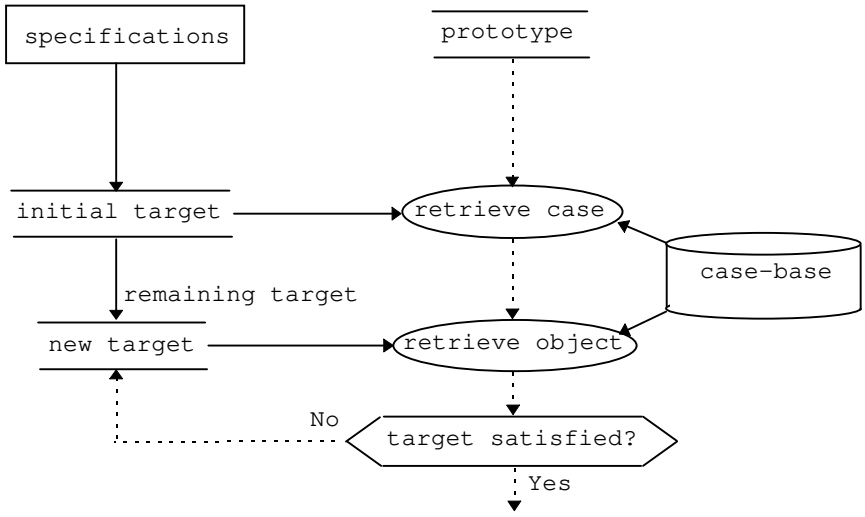

figure 3 Concept Selection

2.3 Concept Modification and Optimisation

The case-base contains designs that have been evaluated and optimised for their originally intended specifications. Parts of these designs can be reused as a starting point near a local or global optimal design. Revision of the initial concepts requires accurate behavioural analysis, heuristic modifications and optimisation.

© Netten, Vingerhoeds, 1996

Retrieved functionality and behaviour cannot provide accurate predictions about behaviour for current specifications. By retrieval of case solutions and substitution of subsolutions, a design is completely instantiated for accurate numerical analysis. Only for the initial concept, resulting from the previous design phase, conflicts in behaviour can be evaluated and criticised. Numerical behavioural results are significantly more accurate and reliable, and retrieved behaviour becomes obsolete.

Initial concepts are revised in two steps: heuristic modification and numerical optimisation. Heuristic modification is necessary for several reasons, see e.g. [Netten and Vingerhoeds '96]. Design variables are symbolic, discrete and continuous. Symbolic variables should be modified before numerical variables are optimised. Heuristics also enable modification towards optimal solutions without extensive numerical search.

3 Case-Based Retrieval and Reuse

Case-based reasoning is applied for the quantification of a prototype solution into an initial conceptual solution. Cases store complete designs with components, evaluated functionality and behaviour. Design cases are obtained from previous design sessions in EADOCS, other CAD systems or literature. The number of cases is small in relation to the number of alternative designs, and the distribution of cases over the design space is strongly biased by previous design intentions.

A case-base of designs rarely contains cases that satisfy completely a design problem. Many different adaptation techniques can be applied to reuse and revise retrieved solutions for a specified design problem [Voβ '96]. Ideally, a design problem is tractable or can be decomposed into tractable subproblems. Each subproblem can be solved independently. A single case solution can be retrieved and adapted, or subsolutions are retrieved from different cases and integrated into a new solution. For most applications, problems are not tractable or accurate adaptations are not available. In [Voβ '96], a variety of different decomposition and adaptation techniques are compared, each developed for a specific or more general application field.

Cases are structured by the objects for the classes of designs, components, functionality and behaviour. Objects are only defined for functionality and behaviour related to the originally design intentions. The solution may satisfy many other functional requirements as well, but this cannot be retrieved from the case itself. Such functionality can only be retrieved from similar prototypical design cases that have been designed for these other requirements. In all other situations, accurate behaviour predictions are required to evaluate a design for other requirements in the third design phase.

© Netten, Vingerhoeds, 1996

The structure of a design solution is represented by the prototype solution, i.e. the components and their configuration. The structure determines component behaviour and design functionality. Combinations of functional requirements determine optimality and feasibility of design solutions and are significant for retrieval and adaptation. However, it is unlikely that the exact combination of case functionality will be specified as a target for retrieval. Many cases will match on some of the target specifications. Multiple cases have to be retrieved and reused to solve a specified set of requirements.

The causal relations between design behaviour and components are not available. This implies that it is not possible to define a target for the retrieval of a component as a subsolution for a specific functional requirement. Retrieval of subsolutions is also restricted by the structure or aggregation of designs. Not every case part can be composed or integrated into a new solution. Adaptation of the structure of a solution can have significant effect on the behaviour for all functional requirements. The structure of the solution is selected by the prototype solution classes. The selected prototype solution is an essential part of the target for retrieval. In EADOCS, the prototype selection phase provides a predefined guidance for retrieval of adaptable solution.

A case memory structure is created to support retrieval by matching and selection of cases and objects. Intervals of attribute values are reused from prototype selection. The memory structure is an extension of the network structure described in [Netten and Vingerhoeds '95]. This classification network is further developed for the domain model, based on the object oriented case structure. Cases are indexed by their objects and subdomains of attribute values. Each attribute interval is an index. Subnetworks are created for each class by categorisation of case objects. The end nodes of each subnetwork categorise each case object. Subnetworks are organised in a collect network according to the class hierarchies. A collect network categorises each case by its objects for functionality, behaviour and solutions. This network has following tasks:

- Partial matching of any object and case with target objects for functionality, behaviour and solutions.
- Filter objects and cases for prototype solutions.
- Determine global similarity for objects and cases for each target separately.
- Determine changes in global similarity for any change in target or prototype.

3.1 Initial Case Retrieval

The target for initial retrieve is a case solution that best satisfies the specifications, and can be reused and revised into the optimal design. Target for retrieval is defined by the set of specified functional requirements and the prototype solution. The source

© Netten, Vingerhoeds, 1996

for retrieval is specified by preferred or prototype solution. The target for initial retrieval is defined by objects for each functional requirement. Global similarity is determined by the matching the pattern of target objects onto objects from cases. Pattern matching is performed within the network structure. The prototype solution classes are bounds on the source, and extend the pattern for matching. Similarity is determined for each case, by summation of similarities between case and target objects.

The best matching case solution is instantiated as the initial solution. This selected case will partially match the target, for two reasons:

1. It is very unlikely that a case exists which satisfies all current specifications. A case matches partially and its similarity measure is reduced.

2. It is very likely that some of the target objects have not been defined for the retrieved case. A case cannot match the target object and receives a similarity = 0.

3.2 Incremental Adaptation

Behaviour of solutions and subsolutions should be predicted accurately for adaptation and integration. Numerical analysis routines can only be applied successfully for initially adapted solutions. Adaptation of a retrieved design for current specifications, or of subsolutions for integration of behaviour, should be postponed until a complete design has been analysed in the concept modification phase. Adaptation of the new solution for unsatisfied target objects is possible by substitution or instantiation of subsolutions from other cases.

Conflicts in integration of components could initially be avoided by retrieval of a complete case solution. The initially retrieved design will match a subset of the target problems. For the remaining targets, minimal adaptations should be retrieved to minimise possible conflicts. The remaining targets, however, cannot be reformulated as new targets for particular case parts and subsolutions. Adaptation of an initial concept requires a different approach to the target formulation for retrieval of subsolutions.

The objective for retrieval and substitution of subsolutions differs from retrieval of an initial solution. In absence of more detailed behavioural models, the assumption can be made that cases with similar solutions will behave similarly as well. Instead of retrieving cases for similar functionality, incremental retrieval for adaptation reverses the process by retrieving cases for similar solutions. The objective is to retrieve the evidence that current solution can also satisfy the remaining targets, or otherwise to retrieve minor adaptations required to solve the remaining target objects. The new target for retrieval of subsolutions can be formulated from following considerations:

© Netten, Vingerhoeds, 1996

- Matching of the retrieved solution for the initial target is identified by the similarity measure. The similarity measure is 0 if no case object is defined for a target. The similarity measure is reduced when the target could not be solved by the new solution.
- Case objects for functionality and behaviour only characterise the intention for original application of a case solution. Similar prototypical solutions will have similar behaviour and functionality. Similarity in functionality and behaviour is implicitly represented in the cases. If similar designs have been applied for different functional requirements, these cases can only be matched by their solution features.
- Minimal adaptation to the conceptual design reduces potential conflicts in interactive behaviour with the initial case solution. Restrictions for minimal adaptations are bounded by the prototype solution. The prototype defines a context for possible adaptations and similarity in behaviour. Case objects from other prototypical solutions cannot be composed into the new solution. Some objects can be similar to objects from the current solution. Differences in attribute values of similar objects, however, will have a much smaller effect on case functionality than the differences in solution structure. The attribute values cannot be related to the remaining targets. The structure of the solution should be maintained if possible. Minor adaptations can be retrieved from cases with almost similar prototypical designs that satisfy a remaining target.

A new case has to be found that satisfies most of the remaining target objects and is most similar to the conceptual design. The new target is formulated by:

- the set of remaining target objects,
- the new conceptual solution, and
- the prototype solution.

Even the best matching case has a slightly different solution than the new solution for the conceptual design. Differences indicate a sufficient adaptation for the new solution to satisfy also a remaining target. The necessity of adaptation for any difference in solution depends also on functionality that is not considered in the new target. Functionality cannot be causally related to differences in solutions. Evaluation of the necessity of adaptation would require accurate analysis of the new solution. A new case is matched only for the remaining target. Differences in other functional requirements are ignored for adaptation.

In the worst situation, a new case cannot satisfy any of the targets already satisfied by the new solution. When such a target is not defined for the new case, this situation cannot be recognised. If it is defined in combination with other differences in

© Netten, Vingerhoeds, 1996

functional requirements, the cause cannot be identified. In most situations, differences in solutions cannot be retrieved for a specific functional requirement. Adaptation should be restricted to only conservative subsolutions that can only improve design behaviour for the remaining target. Conservative adaptations should be defined from domain knowledge, for example from first order approximations of behavioural relations or from design heuristics.

Subsolutions are retrieved from the differences between the new solution and the case solution. Different attribute values of the case can be substituted or objects instantiated for the new solution. Only the conservative adaptations are imposed onto the new solution, and non-conservative adaptations are ignored. Composition of subsolutions is well defined by the data structure for components.

The target objects solved by the new case, are assumed to be solved by the adaptations for the new solution. When the best matching case cannot solve all target objects, the process can be continued for the target objects that still remain. This process is halted when no significantly matching cases can be found for the remaining target objects. Adaptation is continued by modification and optimisation.

4 Example of Concept Selection

Concept selection to quantify and refine the structure of a conceptual design can be exemplified in the following part of a simplified design session. In the design process, at this stage the prototype has been selected. The elaborate specifications are:

Functional specifications:

- Dimensions of the panel: length = 1400mm, width = 400mm.
- 3 loading conditions in the plane of the panel.
    1. (P_x = -5500, P_y = P_{xy} = 0) [N/mm]
    2. (P_y = -2500, P_x = P_{xy} = 0) [N/mm]
    3. (P_{xy} = -2900, P_x = P_y = 0) [N/mm]

    The panel solution must carry each loading condition. The loading conditions are specified with a penalty factor, to express the relative importance of the requirement for feasibility. In this example, all loading conditions with a maximum relative importance.

Selected prototype:

- The panel type is sandwich panel (see Figure 1).
- The sandwich structure is symmetrical around the mid-plane of the core.
- The skin material is carbon (cfrp) or aramid fibre reinforced material (afrp).

© Netten, Vingerhoeds, 1996

Each of the three loading conditions is defined as a separate target problem. The similarity for a target is measured by summation of its category membership. Ideally, the best matching case satisfies each target problem. Usually, as in this example, a case does not satisfy all requirements, nor does it match the required targets.

The set of matching cases is further reduced by matching the selected prototype. The case-base contains 20 sandwich panels. Seven panels satisfy both the selected prototype and one or more of the loading conditions; panel numbers 1, 2, 3, 5, 6, 17 and 18. Five other panels match one or more of the loading conditions, but are discarded for retrieval, as they do not match the selected prototype. Cases 1, 17 and 18 match two out of three conditions.

The user selects panel 17 as the initial conceptual design. Panel 17 matches only the categories for loading conditions 1 and 2. Condition 3 remains as a target for adaptation of the structure or components of panel 17. The selected prototype, as previously specified, is extended with the solution of panel 17.

It is now no longer possible to match on the selected prototype. Only components of the panels 1 and 18 can be retrieved for adaptation, that satisfy the remaining target. The similarity with the 4 components of panel 17 for these panels is measured as 81 % for panel 18 and 69 % for panel 1. Initial adaptation is determined from the differences between solutions of panels 17 and 18 (see Table 1). The outer layers of panel 18 and panel 17 both have 10 layers of aramid cross-ply material. Layer 4 of panel 18 is absent in layer 17. Layers 2 and 3 and the core of panel 18 have fewer plies than panel 17.

Table 1 Conceptual Designs

panel lay-up for all panels			case-base panels		panel 17 optimised	adapted concept	optimised concept
			17	18			
specific mass [gr/cm^2]			2.72	2.71	3.69	3.68	3.42
skin							
layer nr.	θ[1)]	material	number of standard plies in each layer				
1	+45°	afrp	10	10	34	10	10
2	0°	cfrp	22	9	20	22	22
3	90°	cfrp	11	9	8	11	9
4	+45°	cfrp	-	15	-	15	13
core		Al-hc[2)]	7	5	2.5	7	6

[1)] Ply orientation in degrees. [2)] Aluminium Honeycomb material.

© Netten, Vingerhoeds, 1996

Components of panel 18 are reused by substitution or instantiation using conservative adaptations:

- The number of plies is only increased, to avoid loss of stiffness.
- The fibre orientations of the initial plies determine the structure of the panel and are not adapted during this design phase. If necessary, a new layer is added to the laminate lay-up.
- Similar materials are never substituted. A new layer can be added using a different material.

The most significant difference between the two panels is the addition of cross-ply material to resist the increased shear loading P_{xy} of condition 3. Layer 4 of panel 18 is therefore instantiated for panel 17. Reuse of layer 4 from panel 18 results in the adapted concept of Table 1.

To evaluate the improvement of the adaptation, the number of plies of both panel 17 and the adapted panel are numerically optimised for a minimum mass. The number of plies in the outer layer is bounded by a minimum of 10 plies. The optimum panel after adaptation has a relative mass of 3.42 [gr/cm^2]. The optimised panel resulting from panel 17 is restricted to only 4 layers.

5 Conclusions

Applications in EADOCS pose two specific problems for case-based retrieval and reuse:

- Design problems in this application domain cannot be decomposed into tractable subproblems for components.
- Domain knowledge is not available to predict behaviour and suggest adaptations without detailed numerical analysis for a complete design.

Reuse of case subsolutions is necessary to improve the initial solutions for heuristic modification and optimisation. An incremental approach has been presented for adaptation by retrieval of subsolutions. Case reuse is simplified to the substitution of components or attribute values within the structure of the initially retrieved solution. Conflicts resulting from integration of case parts are solved by modification and optimisation in the revision phase.

Targets are defined differently for the retrieval of cases and for incremental adaptation with case parts.

The approach to incremental retrieval can only provide indications for suggesting possible adaptations for remaining functional requirements. Adaptations are

© Netten, Vingerhoeds, 1996

retrieved from similarity in the solutions rather than similarity in the functional requirements. Conditions are defined from domain knowledge by prototype solutions and for conservative adaptation. Initial adaptations can be obtained to improve further modification and optimisation without expensive computations. Adaptations can be suggested that have not been defined as heuristic modifications.

6 References

Bergmann R. and Wilke W., PARIS: Flexible Plan Adaptation by Abstraction and Refinement, ECAI96, Workshop Adaptation in Case-Based Reasoning, 1996.

Börner K., Structural Similarity as Guidance in Case-Based Design, In: *First European Workshop on Case-Based Reasoning EWCBR-93*, M.M. Richter, S. Weβ, K.-D. Althoff, F. Maurer (eds.), SEKI Report SR-93-12, University of Kaiserslautern, 14-19, 1993,.

Hinrichs T.R., Kolodner J.L., 1991, The Roles of Adaptation in Case-Based Design, *Proc. 9th Nat. Conf. on Artificial Intellingence*, Vol. **1**, July 14-19, pp. 28-33, AAAI Press, Menlo Park.

Netten B.D., Vingerhoeds R.A., H. Koppelaar, L. Boullart, Expert Assisted Discrete Optimisation of Composite Structures. In: *European Simulation Symposium 1993*, A. Verbraeck and E.J.H. Kerckhoffs (eds.), Delft, October 25-28,143-148, 1993.

Netten B.D., Knowledge Based Conceptual Design: An Application To Fibre Reinforced Composite Panels. PhD thesis, Faculty of Technical Mathematics and Informatics, Delft University of Technology, (to be submitted 1996).

Netten B.D. and Vingerhoeds R.A., Large-Scale Fault Diagnosis for On-Board Train Systems, *Case-Based Reasoning, Research and Development*, M. Veloso and A. Aamodt (eds.), Lecture Notes in Artificial Intelligence 1010, Springer Verlag, Berlin,.76-76, 1995.

Netten B.D. and Vingerhoeds R.A., Incremental Adaptation for Conceptual Design in EADOCS, ECAI-96 Workshop on Adaptation in Case-Based Reasoning, 1996.

Purvis L. and Pu P., An Approach to Case Combination, ECAI-96 Workshop on Adaptation in Case-Based Reasoning, 1996.

Smyth B. and Keane M.T., Experiments on Adaptation-Guided Retrieval, *Case-Based Reasoning, Research and Development*, M. Veloso and A. Aamodt (eds.), Lecture Notes in Artificial Intelligence 1010, Springer Verlag, Berlin, 313-324, 1995.

© Netten, Vingerhoeds, 1996

Sycara K., Navin Chandra D., Guttal R., Koning J., Narasimhan S., CADET: A Case-Based Synthesis Tool for Engineering Design, *Int. J. Of Expert Systems*, Vol. **4**, No. 2, 157-188, 1992.

Voβ A. (Ed.), Similarity Concepts and Retrieval Methods, FABEL Report No. 13, GMD, Sankt Augustin, 1994.

Voβ A., Exploting previous solutions - made easy, *Engineering Applications of Artificial Intelligence,* Vol. **9** No. 4, to be published (1996).

© Netten, Vingerhoeds, 1996