Exploiting plans, ontologies and constraints in a matchmaker model

Ọdẹ́túnjí. A. Ọdẹ́jobí and Richard J. Wallace
Cork Constraint Computation Centre
University College Cork
Cork, Ireland

Abstract

This paper is intended of provide an exposition to our ongoing research on integrating planning, constraint-based modelling and product ontology in the development of a matchmaker system. The overall aim of the matchmaker is to match a group of attributes of a product to the preferences of a customer. The models for the HCI to support the customer-system interaction, the ontology based knowledge representation and the constraint based reasoning are described. The implementation framework for the model is also discussed. When completed, we expect the matchmaker software to be able to assist buyers and/or support human sellers in the merchandising of a variety of products.

Introduction

Our ongoing matchmaker project has thrown up a number of interesting research challenges in the problem of matching a customer's preference to a particular product. A very common scenario is the situation where the product is described using technical jargon and the customer is expected to determine suitability of such product given his/her preferences. An example of this is the mobile phone product specification. In most mobile phone descriptions available on the internet and product brochures, technical terms such as mega-pixel, anti-blur, roaming, blue tooth, battery life, etc. are used to describe the product's functionally. But many users, particularly those with little or no background technical knowledge of the product, find it difficult to relate these terms to how the product will serve their specific needs. Even when people are familiar with a term, they are not able to relate the technical concept to their preferences. For example, a user may be able to state that s/he wants a camera that will allow him/her to take a photograph that can be expanded by twice the original size without loosing quality. But s/he may not be able to relate this requirement to the technical terms used to describe the camera's resolution, that is the mega-pixel. This issues has been discussed in greater detail elsewhere (Wallace 2002).

Another issue is that of palpability, which relates to the physical experience that users have in respect of the product by way of touching or feeling the products. A display of the product on screen, as is done in most websites and brochures, may not necessarily satisfy the palpability attribute of a product that will be experienced by holding or touching the product. The solution to this aspect of the problem is not very clear at the moment but we think that its solution lies in human psychology and haptic modeling.

When a customer comes into a shop to buy an item of goods, there is often an engagement in the form of a dialogue between the seller and the customer. The seller, who is usually knowledgeable about the nature, varieties and functionalities, of the product will aim to help the customer make his choice. If he is an expert salesman, a quick profile of the customer will assist in this engagement. During the engagement with the sellers, the customer indicates his requirements and the seller tries to match it with the most suitable goods. This scenario can be considered as follows: we have a set of n alternative products $X = \{x_1, x_2, \ldots, x_n\}$, one of which must be chosen. There are two collaborating agents in this scenarios. The seller agent A_s, helps the buyer agent, that is, the customer A_c, to arrive at the final decision about product x_f based on the k attributes $\{x_f^1, x_f^2, \ldots, x_f^k\}$ of products. The overall aim is to match the individual or a group of attributes, to the preference of A_c.

The scenario discussed above generates three important challenges for user computer interaction in such domains:

- the need to express product features and attributes in a form that matches customers perception or conception of its preference value;
- the need to match the user's intention to actions during interaction with the computer;
- the need to give the user the ability to control the selection process of interaction with the computer, including undoing previous actions.

The product model in this context is the abstract description of the product that allows the product features to be accessible for querying during the matchmaking process. The goal is to organise the features-benefit space within a framework that allows customers to be better able to select a product with qualities that optimally satisfy their preference. This is not the classical multi-objective optimisation problem because there is no one-to-one correspondence between product features and customer preference. This

requires a system capable of intelligent behaviour that resembles the reasoning procedure used by a human seller (García-Serrano, Martinez, and Herández 2004; Pu and Faltings 2005). The need for an HCI model naturally emerges in that the system must be able to manage a variety of different coherent reactions.

As the computer is expected to adapt to users rather than the users adapting to the computer, our goal is to model the HCI in such a way that will reduce the potential mismatch between the product's properties and customer's preference or expected preference. We hope to address these challenges by exploiting planning and scheduling, constraint modelling and product ontology techniques within a well integrated modelling framework (Mylonas et al. 2008; Fox and Long 2003).

The range of interaction sequences between customers and the Matchmaker is expected to be broad and to demand a rich representational framework. A plan-based approach would provide a good paradigm for modeling and managing such interaction. If we view a constraint as a specification that serves to define the behaviour of an object, then it is possible to define a constraint-based framework for unifying the different modules of our Matchmaker system. A specification of such a framework will, among other things, define and bound the space within which an acceptable solution lies. We can then exploit the power and generality of the CSP representation to link what are normally thought of as representation features with features that pertain to product content. An advantage for using the CSP as a generalised framework in this context is its power in assisting the derivation of explanations and implications (Freuder, Likitvivatanavong, and Wallace 2001; Feuder, Wallace, and Heffernan 2003), which is a very important attribute that can be exploited for DSS system implementation.

Background

An agent based system for assisting customers in product selection using the CP technique has been proposed in a previous work (Freuder and Wallace 2002). That work offers an alternative to the "Deep Interview" approach, where customers answer queries about product features. This is done by selecting values of these attributes, which can then guide product selection. In the form of dialogue used in that work, the primary mode of interaction is a suggestion (rather than the query) made by the matchmaker to the customer. The customer then replies with a set of corrections, to indicate how the suggested product fails to meet his/her need. The dialogue was modeled as a kind of CSP in which a suggestion corresponds to a solution of a CSP, while a correction specifies a customer constraint that the proposed solution violates. A suggestion that matches the customers requirement is obtained through cycles of such suggestion and correction. The iterative process allows the matchmaker to modify its own CSP to better reflect the customers requirement.

In our ongoing work, we attempt to extend the (Freuder and Wallace 2002) model by addressing two of its inherent shortcomings. The first shortcoming is that the model assumes that the customers comprehend the technical terminologies used for describing products well enough to be able to relate the features to their preference. A second shortcoming, is that the model does not explicitly specify the mode of interaction. We address these shortcoming by using a design product ontology and incorporating a Human Computer Interaction (HCI) module which exploits a planning and scheduling technique. Our goal is to build on the CSP-based model, by using it as the core reasoning component and incorporating an HCI and ontology framework that will improve its decision accuracy. The application of constant-based approach in user computer interaction has been demonstrated (Borning and Duisberg 1986).

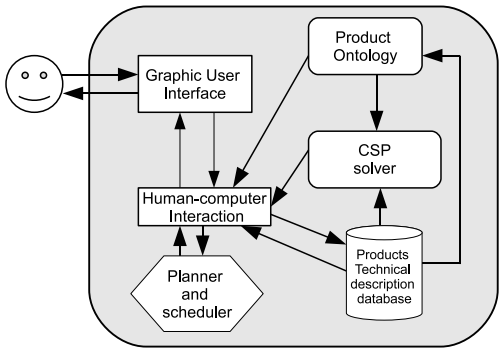

Figure 1: Overview of the Matchmaker system

System architecture

The overview of our Matchmaker system is shown in Figure 1. It comprises six modules which include: (i) graphical user interface (GUI), (ii) human-computer interaction (HCI), (iii) product ontology, (iv) CSP solver, (v) product technical description database server modules and (vi) the planner and scheduler modules. The GUI and HCI modules model the interaction between the seller and customer agents using a plan based approach. The product technical details and ontology servers contain a technical description of the product domain and the taxonomy of the ontology defined over that domain, respectively. The CSP solver implements the reasoning processes in the Matchmaker.

Modelling the process activity

The process activity for the Matchmaker is depicted in Figure 2. When a UI event is detected, the query corresponding to that event is activated and the response provided by the customer is analysed. The analysis will result in the computation of a preference value for that UI event. In the next step, the preference value is then fed into the interaction monitor module. The process automatically issues an action upon the occurrence of the events. The technical features corresponding to the detected events are obtained from the product database and their corresponding contribution to the current preference is evaluated. Those set of questions that are likely to enhance the current preference are then selected from the plan database.

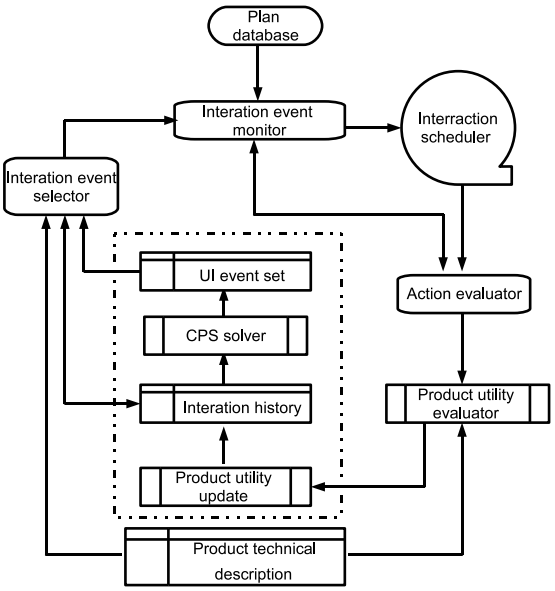

Figure 2: The Matchmaker process activity

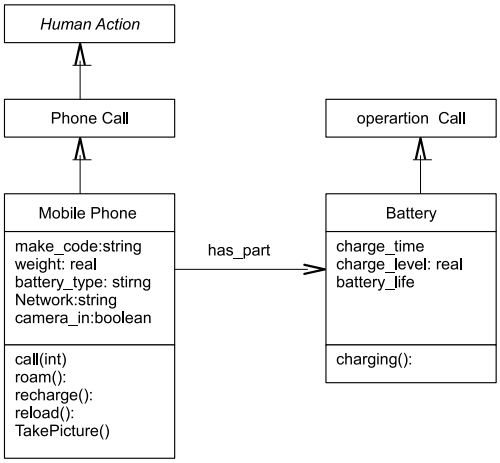

Figure 3: An example representation of class diagram for phone

The product selection are then scheduled for execution. The scheduler generates a set of plans that will make achieving this possible. To facilitate backtracking we included the components within the dotted line (cf. Figure 2). Detected events are stored in the event history and they are fed to the CSP processor during matchmaking. The CSP processor and the UI produces the event sets which are further used by the event detector. This process is repeated until a product that satisfies as many preference as possible is generated or when it is clear that additional query and response session will not increase the preference further.

There is the need to group actions in the UI plan into sets corresponding to different situations in such a way as to yield easier computation of the preference values as well as a more optimal action scheduling.

The ontology module

The basic goals of the ontology module within the Matchmaker architecture are: (i) to allow the system to combine combinatorial inference with user interaction carried out in term that the user is familiar with; (ii) to support a benefits-centred, rather that a feature-centred, approach to product selection. The ontology module contains an ontology of products, and therefore includes any concepts relevant to the description of the characteristics and usage of a particular set of products. As the size of this set is expected to grow as the system develops, we are using an ontology server.

The ontology module is being developed using *Protégé* which is a system that is based on Description Logic. It includes as major components, binary relations between concepts and restrictions regarding the individuals that belong to categories and subcategories. This approach facilitates treating the various features of a product in larger contexts. For example, in the product category $< mobile\ phone >$, a natural associated category is $< phone\ call >$. Us-

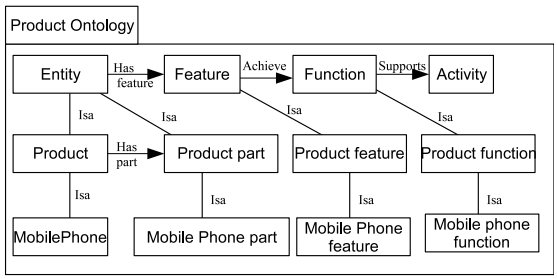

Figure 4: Product ontology diagram

ing the ontology, this can be nested under the concept of $< human\ activity >$, that is an ongoing set of actions, which is $< enabled >$ by certain product $< function >$'s that are in turn $< supported >$ by product $< feature >$'s A typical product class model is shown in Figure 3. The ontology is associated with a database by mean of concept names. The ontology module is still very much a work-in-progress. For example, if the customer in interested in the *roaming* activity, the system presents a window from which a set of features that could facilitate achieving that function.

Planning and scheduling module

The planning module has the dual function of guiding the current user-system interaction and organizing past interactions to form an 'contrived' plan. The purpose of the latter is to provide a relevant context for the user based on previous selections (of activities, features, products, etc.) and to support efficient retracing.

A plan is implemented as a dialog between the customer and the HCI module, and the goal of the dialogue is the article to be selected or purchased. The domain of our problem, therefore, has three active entities, namely:(i) the customer, (ii) the HCI, and (iii) the article to be purchased. This domain can be modeled by way of a problem space representation. As in classical planning, the task is to generate a sequence of actions which, when applied to an initial state, allows the matchmaker to reach a final state. The final state in this case is associated with the recommendation of an acceptable product to a customer and the initial state is associated with the constraints on the functionalities and features of the products.

The planning module is motivated by the need to improve on the "suggestion strategies" of the original (Freuder and Wallace 2002) model. In (Freuder and Wallace 2002), the 'engine' for the procedure was the constraint solver deducing products to select, given the constraints posted in response to user-critiques. The planner seeks to extend this approach by a more precise and extensive definition of states and goals. It also provides a facility for backtracking through the event history component. In this context, a history is a record of user analyses and product related relevance.

The aim of the planning module is to engage the user in a dialogue that guides him/her to select a product that best matches his/her preferences. If F is the set of formal or technical specifications of the product, and R is the function that converts the technical description to an expression of preference, then the aim is to maximise $R(F)$ subject to existing constraints, including non-technical constraints such as price.

The domain described above can be represented by a set of interacting objects, each with different attributes. The technical description of the product, for example, may have a set of $R()$ functions for modelling the value domain, V, and the current values $v_i \in V$ at any point during the matchmaking process.

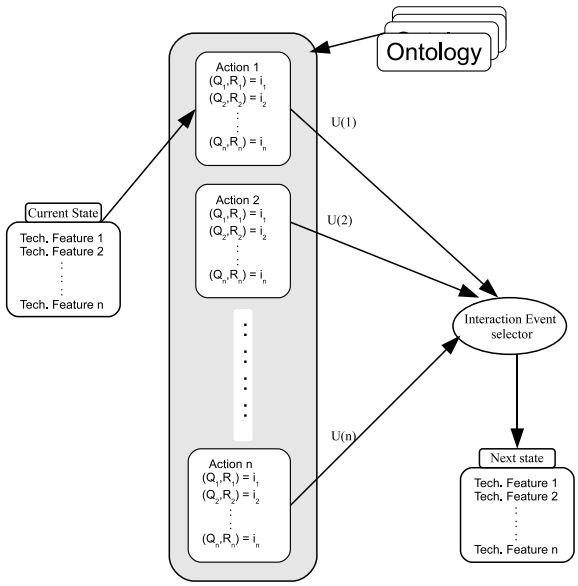

Figure 5: Plan action model to achieve a subgoal

States

We describe the features characterising an item or object of goods as a set $O = \{o_1, o_2, \ldots, o_n\}$ with respective value domains S_i whose current values are $s_i \in S$. A state of the problem world, W, is an element of $S_1 \times S_2 \ldots \times S_n$. At this level of abstraction, objects are basically, "features of a product".

Goal

A goal G of a customer's set of actions A specifies a set of states whose last element describes a situation that satisfies, as much as possible, the set of benefits that the customer expects from the product. We associate the features corresponding to a product in terms of constraints which are then associated with a subgoal $g \in G$. The subgoal is therefore a relation of any two states S^i and S^j.

Action

An action is a plan operator and is modelled as function that uses $R()$ to select next states. The value computed by $R()$ depends on the context of the execution of the action.

Plan

A plan P in this context is defined by the three-tuple: $P =< A, I, G >$, where A is a set of actions, I is a set of feature values that specifies the initial state, and G is the set of goals. The plan can be decomposed into sub-plans that achieve the individual component goals g of the task being modeled. The plan database consists of several loosely coupled sub-plans. Each sub-plan is expected to achieve a well defined task, that is, elicit a preference corresponding to a technical feature of a product. The overall plan is then combined to form a solution to the problem by aggregating the computed preference and executing $R(F)$ maximising some

Plan: The plan algorithm

Data: the planning problem $< A, i \in I, g \in G >$
Result: a solution plan
Associate features to O and each element in S
$S'_j \leftarrow i \in I$
while S'_j does not satisfy g **Do**
call procedure to generate $S^{j\prime}$ from S^j
if procedure aborts
$S'_j \leftarrow next(i)$
endif
endwhile
Return

Figure 6: Plan algorithm

Schedule(Plan): The plan schedule algorithm
$Plan = A$ sequential plan where the actions are sorted by their presentation sequence, i.e. first presentation comes first, followed by the second, etc.
The operator $+$ and $-$ add and remove actions from plan, respectively.

Data:Planning Dbase $PDB = \sum_{1=1}^{n} < A_i, I - i, G_i >$
Result: Schedule of plan
for $i = 1$ to k **Do**
$p_1 = p_i^s$
$R(p_1) =$ previous action before this are added
while (preconditions of A_i not met) **Do**
$p_i \leftarrow$ previous subplan p_r
endwhile
$Plan = Plan + R(p_1)$
endfor
Return

Figure 7: Plan scheduler algorithm

predefined conditions. A method by which sub-plans can be combined to form a complete plan is provided in (Coles et al. 2007). The algorithm for this plan is depicted in Figure 6.

The planning module is designed as an hierarchical planner and it follows a top-down procedure. In this procedure, the general plan is first obtained and then refined, iteratively, until a more specific plan is generated. This approach is adopted because its structure is similar to the ontology we are developing. The planner module, therefore, contains a sequence of actions where every action models a query-response pair corresponding to a feature-preference space. The domain for the plan is expected to be extracted from the ontology.

In the scheduling aspect, interval variables are used to model tasks, logical variables models mutual dependence and product domain ontology are modeled by product classification. We hope to use soft constraint for defining these domains as this will make it possible to relax a constraint if no feasible schedule can be generated. The high level description of the algorithm for the scheduling task is depicted in Figure 7. We assume that the plans corresponding to tasks to be scheduled and their constraints are not known before hand as they are expected to evolve, incrementally, as the matchmaking progresses. When the constraint changes during scheduling and/or execution of all or part of the matchmaking process the respective matchmaking process will be rescheduled. In that case, scheduling is treated as a sequence of constraint satisfaction problems, and commitments are transferred from one problem to the next .

Implementation issues

We are implementing the planning module using the Planning Domain Description Language (PDDL) (Fox and Long 2003). Amongst other thing, this tool facilitates the integration of an ontology based description into a plan. The ontology module is being developed using *Protégé* which is a system that is based on Description Logic. The planning in that context can iterate over the whole ontology and translates the content of the main instances into PDDL process. We hope to exploit constraint-based scheduling (CSB) (Fromherz 2001) as well as the multi-criteria comparison of choice (Ashikhmin and Furems 2005) approaches for the management of choice and sub-plans selection and scheduling. This will allow us to separate the model from the algorithm for the Matchmaker system. In this way we can model a wider variety of constraints, facilitating easier model extension and enabling re-use of the model. The schedular therefore stands between the planning and the execution of an HCI task.

Conclusion

We have provided an exposition to our ongoing work on the development of a matchmaker system. Our approach exploits ontology, planning and scheduling within a constraint-based framework. Important issues that are currently been addressed include the real-time updating of constraints and its integration into the HCI plan and ontology for the matchmaker system. The implementation framework we are using is adequate for realsing the model. We hope that when completed, the matchmaker software will be able to assist buyers and/or support human sellers in the merchandising of a variety of products.

Acknowledgments

The research reported here is supported by Science Foundation Ireland Grant 05/IN/I886 and Marie Curie Grant MTKD-CT-2006-042563

References

Ashikhmin, I., and Furems, E. 2005. UniComBOS- intelligent decision support system for multi-criteria comparison and choice. *Journal of Multi-Criterial Decision Analysis* 13:147–157.

Borning, A., and Duisberg, R. 1986. Constraint-based tools for building user interface. *ACM Transactions on Graphics* 5(4):345–374.

Coles, A.; Fox, M.; Long, D.; and Smith, A. 2007. Planning with respect to an existing schedule of events. In Boddy, M.; Fox, M.; and Thiébaux, S., eds., *Proc. of* 7^{th} *International Conference on Automated Planning and scheduling*, 81–88.

Feuder, E. C.; Wallace, R. J.; and Heffernan, R. 2003. Ordinal constraint satisfaction. In 5^{th} *International Workshop on Soft Constraints -SOFT'03*.

Fox, M., and Long, D. 2003. PDDL2.1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research* 20:61–124.

Freuder, E. C., and Wallace, R. J. 2002. Suggestion strategies for constraint-based matchmaker agents. *International Journal on Artificial Intelligence Tools* 11(1):3–18.

Freuder, E. C.; Likitvivatanavong, C.; and Wallace, R. J. 2001. Deriving explanations and implication for contraint satisfaction problems. In Walsh, T., ed., *Principles and Practice of Contriant Programming- CP 2001 BRCIM 2002*, volume 2239 of *Lecture Note in Computer Science*, 585–589.

Fromherz, M. P. J. 2001. Constraint-based scheduling. In *Invited Tutorial Paper, The American Control conference ACC'01*. Arlington, VA: www.parc.com.

García-Serrano, A. M.; Martinez, P.; and Herández, J. Z. 2004. Using AI techniques to support advanced interaction capabilities in virtual assistant for e-commerse. *Expert System with Application* 26:413–426.

Mylonas, P. H.; Vallet, D.; Castells, P.; Fernández, M.; and Avrithis, Y. 2008. Personalized information retrieval based on context and ontological knowldge. *The Knowledge Engineering Review* 23(1):73–100.

Pu, P., and Faltings, B. 2005. Intelligent interface for preference-based search. In Riedl, J.; Jameson, A.; Billsus, D.; and Lau, T., eds., *International Conference on Intelligent User Interfaces*.

Wallace, R. J. 2002. Contraint-based matchmaking: A personal (interim) perspective. In *Joint ERCM CologNet Workshop CSCLP 2002*, 169–179.