Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services

Markus Zanker[1,2], Markus Aschinger[1], and Markus Jessenitschnig[3]

[1] University Klagenfurt, Austria
markus.zanker@uni-klu.ac.at
[2] ConfigWorks Informationssysteme & Consulting GmbH, Austria
masching@edu.uni-klu.ac.at
[3] eTourism Competence Center Austria (ECCA)
markus.jessenitschnig@etourism-austria.at

Abstract. The composition of product bundles like tourism packages, financial services portfolios or compatible sets of for instance skin care products is a synthesis task that requires knowledgeable information system support. We present a constraint-based Web configurator capable of solving such tasks in e-commerce environments. Our contribution lies in hybridizing a knowledge-based configuration approach with collaborative methods from the domain of recommender systems in order to guide the solving process by user preferences through large product spaces. The system is implemented on the basis of a service-oriented architecture and supports a model-driven approach for knowledge acquisition and maintenance. An evaluation gives acceptable computation times for realistic problem instances.

1 Introduction

In many e-commerce situations consumers are not looking for a single product item but they require a set of several different products. For instance on e-tourism platforms online users may either selectively combine different items like accommodation, travel services, events or sights - or they might choose from an array of pre-configured packages. When bundling different items on their own users are performing a synthesis task comparable to configuration.

Mittal and Frayman [1] defined configuration as a special type of design activity, with the key feature that the artifact being designed is assembled from a set of pre-defined components. When bundling different products with each other, product categories represent these pre-defined components. Additional knowledge is required that states which combinations of components are allowed and which restrictions need to be observed. For instance, proposed leisure activities should be within reach from the guest's accommodation or recommended sights need to be appropriate for kids if the user represents a family. Nevertheless, the problem of computing product bundles that are compatible with a set of domain constraints differs from traditional configuration domains in the sense that

B. Benatallah et al. (Eds.): WISE 2007, LNCS 4831, pp. 273–284, 2007.

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

fewer restrictions apply. For instance a car configurator computes a valid vehicle variant satisfying the user's requirements and all applicable commercial and technical restrictions that derive from the manufacturer's marketing and engineering experts. Contrastingly, when configuring a product bundle relatively fewer strict limitations will apply, because not a new materialized artifact is created but an intangible composition of products or services is defined. As a consequence an order of magnitude more combinations of components are possible and the question of finding an optional configuration becomes crucial.

Optimality can be either interpreted from the provider perspective, e.g. the configuration solution with the highest profit margin, or from the customer perspective. In the latter case the product bundle that best fits the customer's requirements and preferences is proposed. Recommender systems are intended to derive a ranked list of product instances following an abstract goal such as maximizing user's utility or online conversion rates [2]. For instance collaborative filtering is the most common recommendation technique. It exploits clusters of users that showed similar interest in the past and proposes products that their statistically nearest neighbors also liked [3]. We exploit this characteristic of recommender systems for deriving a personalized preference ordering for each type of product in our configuration problem. Our contribution thus lies in integrating a constraint-based configuration approach with soft preference information that derive from recommender systems based on for instance the collaborative filtering paradigm. This way we can guide the solving process towards finding optimal product bundles according to the users assumed preference situation.

Contrasting the work of Ardissono et al. [4] and of Pu and Faltings [5] we do not require explicit preference elicitation by questioning or example-critiquing, but depending on the underlying recommendation paradigm community knowledge and past transaction data are utilized.

The paper is structured as follows: First we give an extensive elaboration on related work and then present a motivating example for illustration purposes in Section 3. Furthermore, we give details on the system development in Section 4 and finalize with practical experiences and conclusions.

2 Related Work

Configuration systems are one of the most successful applications of AI-techniques. In industrial environments they support the configuration of complex products and services and help to reduce error rates and throughput time significantly compared to manual processes. Depending on the underlying knowledge-representation mechanism rule-based, model-based and case-based frameworks for product configuration exist [6]. Configurators that build on the constraint satisfaction problem (CSP) paradigm are within the family of model-based approaches [7,8]. They clearly separate between an explicitly represented knowledge base and a problem solving strategy. In technical domains such as telephone switching systems large problem instances with tens of thousands of components exist. Efficient solving strategies exploit the functional decomposition of

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

the product structure to determine valid interconnections of the different components with each other [7]. Pure sales configuration systems such as online car or pc configuration systems[1] are much simpler from the computational point of view. They allow their users to explore the variant space of different options and add-ons and ensure that users place orders that are technically feasible and have a correct price. However, these systems are typically not personalized, i.e. they do not adapt their behavior according to their current user.

The CAWICOMS project was among the first to address the issue of personalization in configuration systems [4]. They developed a framework for a personalized, distributed Web-based configurator. They build on dynamic user interfaces that adapt their interaction style according to abstract user properties such as experience level or needs situation. The system decides on the questioning style (e.g. asking for abstract product properties or detailed technical parameters) and computes personalized default values if the user's assumed expertise is not sufficient for answering. Pu and Faltings [5] present a decision framework based on constraint programming. They show how soft constraints are well-suited for supporting preference models. Their work concentrates on explicitly stated user preferences especially via an example critiquing interaction model. This way tradeoff decisions are elicited from users. Given a specific product instance users may critique on one product property and specify on which other properties they would be willing to compromise. Soft constraints with priority values are revised in such an interaction scenario and guide the solution search.

In contrast to the work in [4,5], we do not solely rely on explicitly stated user feedback, but integrate configuration with recommender systems to include assumed user preferences.

Recommender systems constitute a base technology for personalized interaction and individualized product propositions in electronic commerce [2]. However, they do not support synthesis tasks like configuration. Given sets of items and users, recommender systems compute for each single user an individualized list of ranked items according to an abstract goal such as degree of interest or buying probability [2]. Burke [9] differentiates between five different recommendation paradigms: *collaborative*, *demographic* and *content-based* filtering as well as *knowledge* and *utility-based* recommendation. Collaborative filtering is the most well known technique that utilizes clusters of users that showed similar preferences in the past and recommends those items to a user her cluster neighbors have liked [3,10]. Content-based filtering records those items the user has highly rated in the past and proposes similar ones. Successful application domains are for instance news or Web documents in general, where the system learns user preferences in the form of vectors of term categories [11]. Demographic filtering builds on the assumption that users with similar social, religious or cultural background share similar views and tastes. Knowledge- and utility-based methods rely on a domain model of assumed user preferences that is developed by a human expert. Jannach [12] developed a sales advisory system that maps explicitly stated abstract user requirements onto product characteristics and computes

[1] For instance, see http://www.bmw.com or http://store.apple.com

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

a set of best matching items. Case-based recommender systems exploit former successful user interactions denominated as cases. When interacting with a new user, the system retrieves and revises stored cases in order to make a proposition. Thus, a human expert is required to define efficient similarity measures for case retrieval [13]. Burke [14] and Ricci [15] have researched and developed several systems within this field.

We integrated our configurator with a polymorphic recommendation service that can be instantiated with an arbitrary recommendation paradigm [16]. Details will be given in Section 4.

Preference-based search takes a more interactive and dynamic approach towards personalized retrieval of items. They build and revise the preference model of the user during interaction, instead of having it beforehand. One of the first applications of interactive assessment of user preferences was the Automated Travel Assistant [17]. Further extensive works on interactive preference elicitation has been conducted recently [18,19,20] and [21] includes an extensive overview.

The work on preference-based search is orthogonal to our contribution. Our implementation supports interactivity between the system and the user during exploration of the search space. Therefore, additional preference constraints can be added and revised at each round of interaction (see Subsection 4.4).

3 Motivating Example

We start the description of our approach by giving a motivating example. Figure 1 depicts a service configuration scenario from the e-tourism domain. The user model states a set of specific requirements for *John* like a travel package to the city of Innsbruck or that the solution should be appropriate for a family with children. In addition, contextual parameters like the weather situation or the current season are represented in the system context. Concrete product instances and their evaluations are part of the Product model, e.g. sights or restaurants. The dotted arrows exemplify some constraints of the configuration knowledge base like *The location of the Sight/Restaurant/Event and the location of the Accommodation should be the same* or *If the weather outlook is rainy propose an indoor event.* Additional preference information is included into the configuration knowledge base by integrating recommendation services for each class of products: for instance, when collaborative filtering recommends items from the product class *Event*, transaction records of other users and the interaction history of *John* will be exploited. Consequently, the higher ranked items in the recommendation result are more probable to be included in the configuration solution.

Informally, given a User Model, a System context and a set of constraining dependencies, the task of the system is to find a set of products from different categories that is optimal with respect to the preference information derived from external recommender systems. In the following we will detail the development of the system.

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

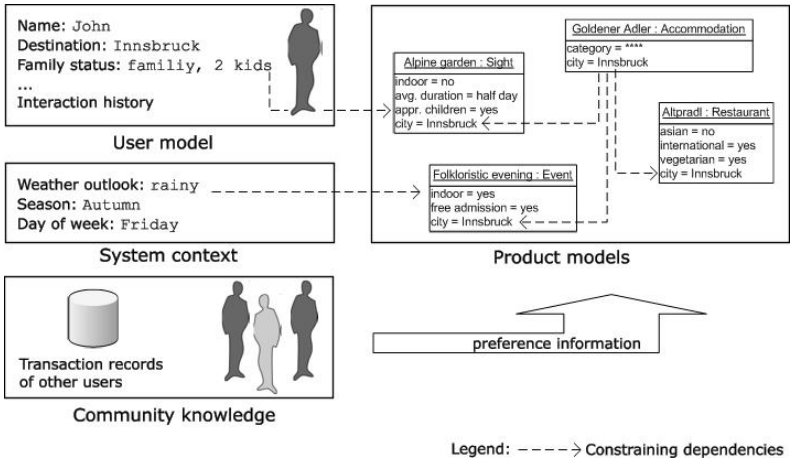

Fig. 1. Example scenario

4 Development

When designing the system we decided on the constraint programming paradigm for knowledge representation and problem solving - comparable to most of the configuration systems referenced in Section 2. The *Choco* Open Source constraint library [22] in Java is the basis for our implementation. In the next subsections we will describe the constraint-representation of the domain model, aspects of knowledge acquisition as well as our system architecture.

4.1 Architecture

In Figure 2 we sketch the system architecture. It consists of a configuration service component and several recommender services delivering personalized rankings of instances from a given class of products. We realized a service-oriented architecture that supports communication via Web services, a php-API and a Java-API. This enables flexible integration with different Web applications, distributed deployment scenarios and ensures the extensibility towards additional recommendation services. The latter requires sharing the identities of users and product instances as well as the semantics of user and product characteristics among all components. This is realized by a central user and product model repository offering service APIs, too.

The user interacts with a Web application, that itself requests personalized product bundles from the configurator via the service-API. The evaluation of contextual parameters can be requested by the same communication means. We have implemented variants of collaborative and content-based filtering recommenders as well as utility and knowledge-based ones as sketched in [16]. A more detailed evaluation of different recommendation strategies on commercial data was done in [23]. Next, we will detail on the domain model for the configuration service itself.

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

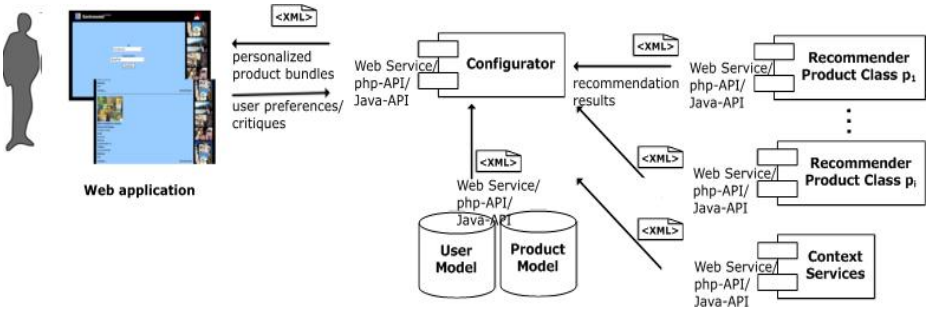

Fig. 2. System architecture

4.2 Model Representation

The constraint satisfaction paradigm is employed for knowledge representation. A Constraint Satisfaction Problem (CSP) is defined as follows [24]:

A CSP is defined by a tuple $\langle X, D, C\rangle$, where $X = \{x_1, \ldots, x_n\}$ is a set of variables, $D = \{d_1, \ldots, d_n\}$ a set of corresponding variable domains and $C = \{c_1, \ldots, c_m\}$ a set of constraints.

Each variable x_i may only be assigned a value $v \in d_i$ from its domain. A constraint c_j further restricts the allowed assignments on a set of variables. On each partial value assignment to variables it can be determined if a constraint is violated or not. In addition, all constraints $c_j \in C$ are defined to be either *hard* ($c_j \in C_{hard}$) or *soft* ($c_j \in C_{soft}$), where $C = C_{hard} \cup C_{soft}$ and $C_{hard} \cap C_{soft} = \emptyset$. Soft constraints may be violated. Each of them is associated with a penalty value and the sum of penalty values of violated constraints has to be minimized when looking for an optimal solution. For further details and definitions of CSPs we refer to [24].

Next, based on this formalization we present our domain model. It consists of a tuple $\langle U, P, X_{UM}, X_{Cx}, X_{PM}, D_{PM}, C_{hard}, C_{soft}\rangle$, where:

- $U = \{u_1, \ldots, u_n\}$ is a set of users,
- $P = \{p_1, \ldots, p_i\}$ a set of product classes - each is associated with a recommendation service that delivers a personalized item ranking upon request,
- $weight(p_j)$ the relative weight of product class p_j used in the overall optimization function,
- $X_{UM} = \{x_1, \ldots, x_j\}$ a set of variables from the User model,
- $X_{Cx} = \{x_1, \ldots, x_k\}$ a set of variables modeling the system context,
- $X_{PM} = \{p_1.x_1, \ldots, p_1.x_m, \ldots, p_i.x_1, \ldots, p_i.x_o\}$ a set of variables modeling product properties, where $p.x$ denotes the product property x of product class p and $p[j].x$ the concrete evaluation of x for product instance j,
- $D_{PM} = \{p_1.d_1, \ldots, p_1.d_m, \ldots, p_i.d_1, \ldots, p_i.d_o\}$ a set of corresponding domains for product properties,
- $C_{hard} = \{c_1, \ldots, c_p\}$ a set of hard constraints on variables in $X = X_{UM} \cup X_{Cx} \cup X_{PM}$,
- $C_{soft} = \{c_1, \ldots, c_q\}$ a set of soft constraints on variables in X and finally
- $pen(c_j)$ the penalty value for relaxing soft constraint c_j.

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

This domain model is defined and maintained by domain experts themselves in order to reduce the traditional knowledge acquisition bottleneck as outlined in the next subsection.

4.3 Model Definition and CSP Generation

Model definition and maintenance is supported by a modular editor environment based on the Eclipse RCP (Rich-Client Platform) technology[2]. Figure 3 gives an impression of the interaction with the knowledge acquisition workbench. First, those user characteristics that are relevant for solving are retrieved from the User model repository. Second, additional external services providing contextual data such as the current season of the year or weather information are selected. In a third step, the set of product classes P and their associated recommender services are integrated. For each class of products relevant properties are selected from the underlying repository. Finally, hard and soft constraints are defined using a context-sensitive editor.

In Figure 4 we depict the whole process. It is separated into a design phase, where the model is defined and maintained, and an execution phase. During the latter a specific user u is given, when the configurator is invoked. First, product rankings are retrieved from recommendation services and then corresponding product characteristics are requested from the product model repository. In the

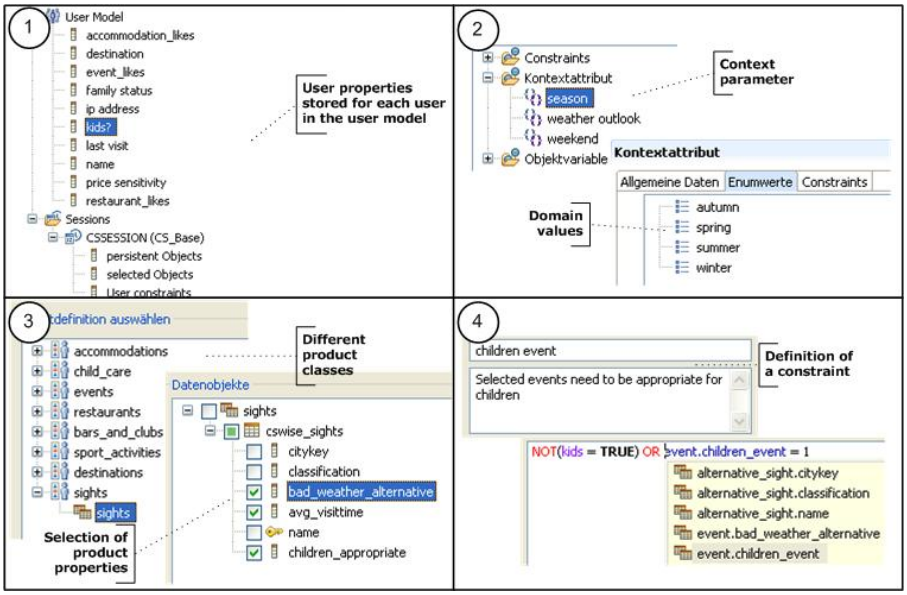

Fig. 3. Knowledge acquisition workbench

[2] See http://www.eclipse.org/rcp for reference

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

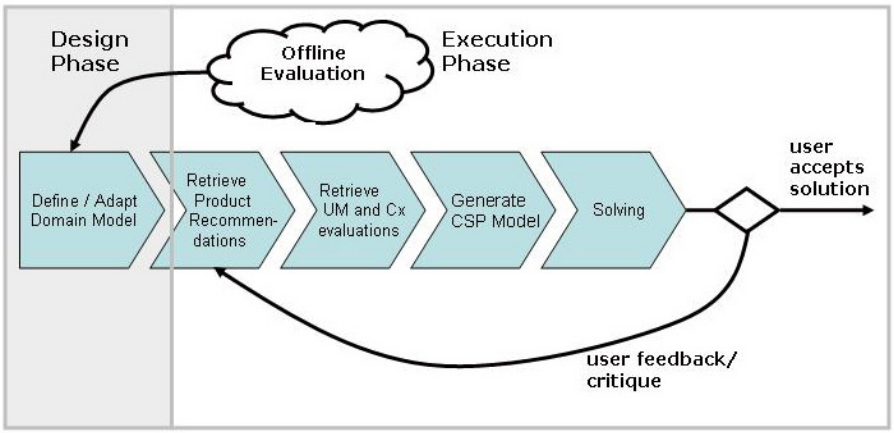

Fig. 4. Design and Execution phase

next step, all variables in $X_{UM} \cup X_{Cx}$ are assigned values by requesting the User model repository and the context services.

Then a CSP model is generated on the fly and the following transformation steps are done:

- Create all variables in $X_{UM} \cup X_{Cx}$ in the CSP model and assign them their respective evaluations.
- For each product class $p \in P$, $p[1]$ denotes the highest ranked product instance and $p[p_n]$ the lowest ranked one. For all $p \in P$ we create an index variable $p.idx$ with the domain $p.d_{idx} = \{1, \ldots, p_n\}$. The index represents the preference information on the instances of a product class. If two product instances fulfill all constraints then the one with the lower index value should be part of the recommended bundle.
- Create all variables in X_{PM} and assign them domains as follows: $\forall p \in P \forall p.x \in X_{PM} \forall i \in p.d_{idx} \quad p[i].x \in p.d_x$, i.e. for a given product property x of class p all evaluations of recommended instances need to be in its domain $p.d_x$.
- Furthermore, integrity constraints are required to ensure that the value assigned to the index variable of product class p is consistent with the values assigned to its product properties $p.x$: $\forall p \in P \forall p.x \in X_{PM} \forall i \in p.d_{idx} \quad p.idx = i \rightarrow p.x = p[i].x$. Hence product instances are modeled, although the *Choco* CSP formalism does not support object orientation.
- Insert all domain constraints from $C_{hard} \cup C_{soft}$.
- For each soft constraint $c \in C_{soft}$, create a variable $c.pen$ that holds the penalty value $pen(c)$ if c is violated or 0 otherwise.
- Create a resource variable res and a constraint defining res as the weighted sum of all variables holding penalty values for soft constraints and all weighted index variables. The tradeoff factor between soft constraints and

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

product class indexes as well as all weights can be adapted via the knowledge acquisition workbench.

4.4 CSP Solving

Once the CSP model is generated, the *Choco* solver is invoked utilizing its optimization functionality. The goal is to find an assignment to all variables in the CSP model that does not violate any hard constraint and minimizes the resource variable res. We extended the branch and bound optimization algorithm [22] to compute the top n solution tuples instead of solely a single product bundle. The solver is capable of following two different strategies for computing n product bundles, namely *1-different* and *all-different*. *1-different* signifies that each tuple of product instances in the set of n solutions contains of at least one product instance that is different in all other solution bundles. In contrast *all-different* means that the intersection between two product bundles from the set of solutions is empty, i.e. a product instance may only be part of at most one solution tuple. In the next section we report on computation times for CSP generation and solving.

5 Evaluation

Based on our e-tourism application domain we developed an example scenario consisting of 5 product classes with a total of 30 different product properties. Their domains are string, bounded integer and boolean. We defined a total of 23 representative domain constraints (13 hard and 10 soft constraints) for configuration knowledge.

We varied the number of the requested product instances from recommender services in 5 different steps between 5 and 100 and denoted the resulting CSP models M1 to M5. Details on problem sizes and times for 'on-the-fly' generation of CSP models are given in Table 1. As can be seen, the number of variables does not depend on the number of recommendations and stays the same for all models. However, the average size of variable domains increases from M1 to M5 due to the higher amount of product instances per product class. The number of constraints depends mainly on the aforementioned integrity constraints to model product instances. Therefore, M5 contains ten times more constraints than M1.

Table 1. Model sizes used in evaluation

Model	Number of Recommendations	Number of Vars	Average Domain Size	Number of Constraints	Generation time in ms
M1	5	58	7,45	206	10
M2	10	58	8,73	374	20
M3	30	58	13,55	1010	60
M4	50	58	16,5	1355	95
M5	100	58	23,23	2093	135

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

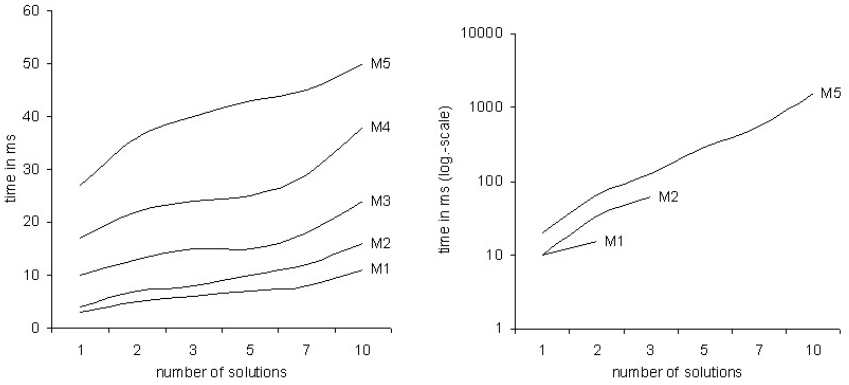

Fig. 5. Results for solution strategies *1-different* (left) and *all-different* (right)

As can be seen in Table 1, the times for generating even the large M5 model are acceptable for interactive applications.

In order to evaluate the performance of our system we ran the experiments on a standard Pentium 4 3GHz processor. The time measurements for the solving step are depicted in Figure 5. For each model we ran the experiment for 100 users and averaged times. We evaluated both strategies to compute a set of top n solution tuples as well as varied the number of n between 1 and 10. The solving times (y-axis) for the *all-different* strategy are on a logarithmic scale. Moreover, we skipped graphs for M3 and M4 for better readability. The *1-different* strategy is less complex and therefore solving time does not exceed 50 ms. To the contrary, *all-different* requires significantly longer computation times. Nevertheless, the performance is still satisfactory. The highest value for model M5 is about 1.5 seconds with 10 solutions computed for a reasonable problem in our domain. Although in typical e-commerce situations at most 10 different product bundles will be required, requests for 100 solution tuples can still be handled within a satisfying time range. For instance, when using the *1-different* solution strategy the solver required around 220 ms to calculate the top 100 solutions for model M5. In case of *all-different* 15 solutions are the maximum amount of solutions we could find within the same model. It took us 6 seconds which indicates the bottleneck of an interactive system.

Nevertheless, we have evidence to believe that our integration of a configurator with different recommender systems is efficient enough to solve standard product bundling tasks in online sales situations. Further experiments in different example domains will be conducted.

6 Conclusion

We presented the development of a generic Web configurator that is realized by integrating recommendation functionality with a constraint solver. Thus, our

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

contribution lies in a novel strategy to personalize configuration results on product bundles. The system observes on the one hand explicit domain restrictions and on the other hand user preferences deriving from recommender systems. The application is developed within the scope of an industrial research project in the e-tourism domain. Our evaluation on realistic problem sizes showed acceptable computation times and system deployment is planned for the next months.

References

1. Mittal, S., Frayman, F.: Toward a generic model of configuration tasks. In: 11^{th} International Joint Conferences on Artificial Intelligence, Menlo Park, California, pp. 1395–1401 (1989)
2. Adomavicius, G., Tuzhilin, A.: Towards the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6) (2005)
3. Resnick, P., Iacovou, N., Suchak, N., Bergstrom, P., Riedl, J.: Grouplens: An open architecture for collaborative filtering of netnews. In: Computer Supported Collaborative Work (CSCW), Chapel Hill, NC (1994)
4. Ardissono, L., Felfernig, A., Friedrich, G., Goy, A., Jannach, D., Petrone, G., Schäfer, R., Zanker, M.: A framework for the development of personalized, distributed web-based configuration systems. AI Magazine 24(3), 93–108 (2003)
5. Pu, P., Faltings, B.: Decision tradeoff using example-critiquing and constraint programming. Constraints 9, 289–310 (2004)
6. Sabin, D., Weigel, R.: Product configuration frameworks - a survey. IEEE Intelligent Systems 17, 42–49 (1998)
7. Fleischanderl, G., Friedrich, G., Haselböck, A., Schreiner, H., Stumptner, M.: Configuring large systems using generative constraint satisfaction. IEEE Intelligent Systems 17, 59–68 (1998)
8. Mailharro, D.: A classification and constraint-based framework for configuration. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 12, 383–397 (1998)
9. Burke, R.: Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction 12(4), 331–370 (2002)
10. Pazzani, M.: A framework for collaborative, content-based and demographic filtering. Artificial Intelligence Review 13(5/6), 393–408 (1999)
11. Balabanovic, M., Shoham, Y.: Fab: Content-based, collaborative recommendation. Communications of the ACM 40(3), 66–72 (1997)
12. Jannach, D.: Advisor suite - a knowledge-based sales advisory system. In: de Mantaras, L.S.L. (ed.) PAIS. 16^{th} European Conference on Artificial Intelligence - Prestigious Applications of AI, pp. 720–724. IOS Press, Amsterdam (2004)
13. O'Sullivan, D., Smyth, B., Wilson, D.: Understanding case-based recommendation: A similarity knowledge perspective. International Journal of Artificial Intelligence Tools (2005)
14. Burke, R.: The Wasabi Personal Shopper: A Case-Based Recommender System. In: IAAI. 11^{th} Conference on Innovative Applications of Artificial Intelligence, Trento, IT, pp. 844–849. AAAI, USA (2000)
15. Ricci, F., Werthner, H.: Case base querying for travel planning recommendation. Information Technology and Tourism 3, 215–266 (2002)

Zanker M., Aschinger M., Jessenitschnig M.: Development of a Collaborative and Constraint-Based Web Configuration System for Personalized Bundling of Products and Services. In: B. Benatallah et al. (Hrsg.): 8th International Conference on Web Information Systems Engineering, Nancy, France., 2007 (LNCS, 4831), pp. 273-284.

16. Zanker, M., Jessenitschnig, M.: Iseller - a generic and hybrid recommendation system for interactive selling scenarios. In: ECIS. 15^{th} European Conference on Information Systems, St. Gallen, Switzerland (2007)
17. Linden, G., Hanks, S., Lesh, N.: Interactive assessment of user preference models: The automated travel assistant. In: UM. 5^{th} International Conference on User Modeling, Lyon, France (1997)
18. Shimazu, H.: Expert clerk: Navigating shoppers' buying process with the combination of asking and proposing. In: IJCAI. 17^{th} International Joint Conference on Artificial Intelligence, pp. 1443–1448 (2001)
19. Smyth, B., McGinty, L., Reilly, J., McCarthy, K.: Compound critiques for conversational recommender systems. In: IEEE/WIC/ACM International Conference on Web Intelligence (WI), pp. 145–151. IEEE Computer Society, Washington, DC, USA (2004)
20. Viappiani, P., Faltings, B., Pu, P.: Evaluating preference-based search tools: a tale of two approaches. In: AAAI. 22^{th} National Conference on Artificial Intelligence (2006)
21. Viappiani, P., Faltings, B., Pu, P.: Preference-based search using example-critiquing with suggestions. Artificial Intelligence Research 27, 465–503 (2006)
22. Laburthe, F., Jussien, N., Guillaume, R., Hadrien, C.: Choco Tutorial, Sourceforge Open Source, `http://choco.sourceforge.net/tut_base.html`
23. Zanker, M., Jessenitschnig, M., Jannach, D., Gordea, S.: Comparing recommendation strategies in a commercial context. IEEE Intelligent Systems 22, 69–73 (2007)
24. Tsang, E.: Foundations of Constraint Satisfaction. Academic Press, London, UK (1993)