Configuration Knowledge Representation and Reasoning

Lothar Hotz [§], Alexander Felfernig[*], Markus Stumptner[†], Anna Ryabokon[‡], and Claire Bagley[+]

[*]Graz University of Technology, Graz, Austria
[‡]University of Klagenfurt, Klagenfurt, Austria
[†]University of South Australia, Adelaide, Australia
[+]Oracle Coropration, USA
[§]HITeC e.V., University of Hamburg, Hamburg, Germany

Contents

- Constraint Satisfaction Problems
  - Static Constraint Satisfaction
  - Dynamic Constraint Satisfaction
  - Generative Constraint Satisfaction
- Graphical Knowledge Representations
  - Feature Models
  - UML Configuration Models
  - Formalization of UML Configuration Models

Constraint Technologies

„Constraint technologies are one of the closest approaches computer science has yet made to the Holy Grail of programming: a user states the problem, the computer solves it"

[Freuder, 1997]

Constraint Satisfaction Problem (CSP)

Definition (Constraint Satisfaction Problem – CSP). A constraint satisfaction problem (CSP) can be defined by a triple (V, D, C) where V is a set of finite domain variables {*v*1, *v*2, . . . , *vn*}, D represents variable domains {*dom(v*1*), dom(v*2*), . . . , dom(vn)*}, and C represents a set of constraints defining restrictions on the possible combinations of variable values ({*c*1, *c*2, . . . , *cm*}).

Solution for a CSP

Definition (CSP Solution). A solution for a given CSP = (V, D, C) is represented by an assignment $S = \{ins(v1), ins(v2), . . . , ins(vn)\}$ where $ins(vi) \in dom(vi)$. S is required to be *complete*; that is each variable of the CSP definition has a value in S and is consistent (i.e., S fulfills the constraints in C).

Configuration Task

Definition (Configuration Task). A configuration task can be defined as a CSP (V, D, C) where $V = \{v1, v2, \ldots, vn\}$, $D = \{dom(v1), dom(v2), \ldots, dom(vn)\}$, and $C = CKB \cup REQ$. *CKB* represents the configuration knowledge base (the configuration model) and *REQ* represents a set of user (customer) requirements.

Configuration (Solution)

Definition (Configuration). A configuration (solution) S for a given configuration task (V, D, *CKB* ∪ *REQ*) is represented by an assignment *S* = {*ins(v*1*), ins(v*2*), . . . , ins(vn)*} where *ins(vi)* ∈ *dom(vi)* and S is complete and consistent with the constraints in *CKB* ∪ *REQ*.

A Simple Configuration Task: Map Coloring

All regions $y \neq x$ that are direct neighbors of x must have a different color (different from the color of x)

Corresponding Configuration Task

V = {WA, NT, SA, Q, NSW, V, T}

D = {dom(WA)={r,g,b}, dom(NT)={r,g,b}, dom(SA)={r,g,b}, dom(Q)= {r,g,b}, dom(NSW)={r,g,b}, dom(V)={r,g,b}, dom(T)={r,g,b}}

CKB = {WA $\neq$ NT, WA $\neq$ SA, NT $\neq$ SA, NT $\neq$ Q, SA $\neq$ Q, SA $\neq$ NSW, SA $\neq$ V, Q $\neq$ NSW, NSW $\neq$ V}

$REQ = \{WA = r\}$

Graphical CSP Representation

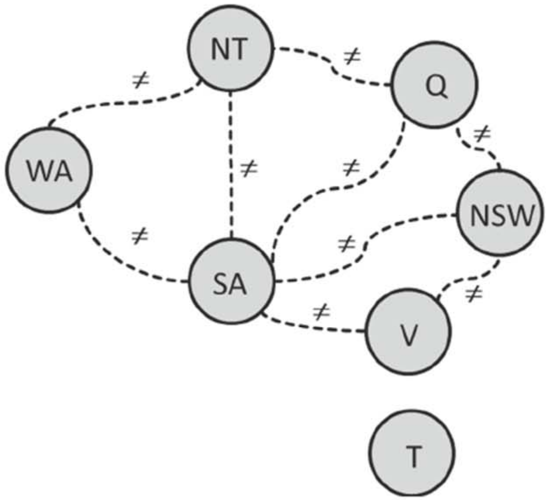


FIGURE 6.2

Map coloring configuration model: graphical representation of a CSP where the nodes represent the variables and the arcs represent constraints.

Graphical CSP Representation

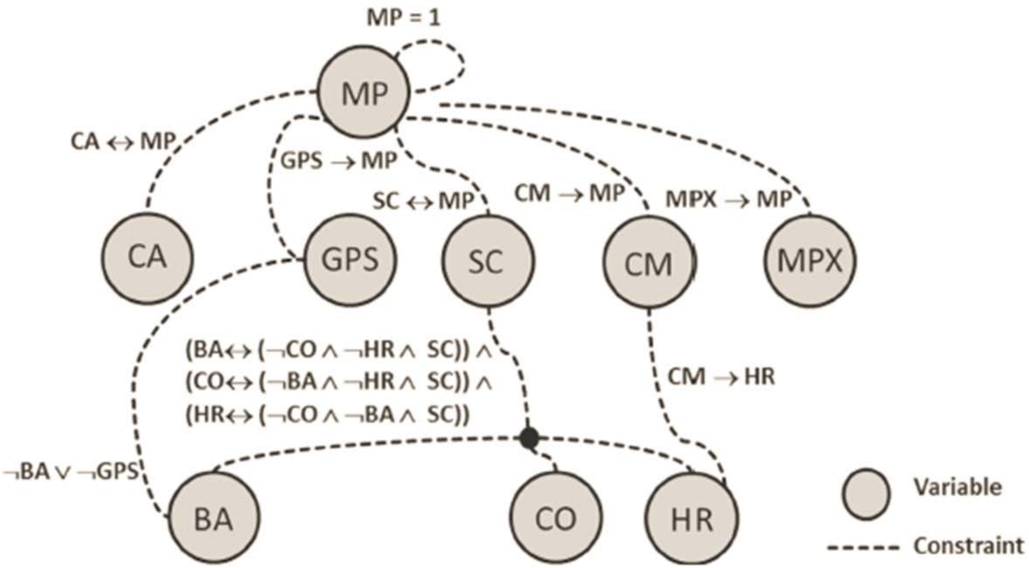


FIGURE 6.3

Simple Mobile Phone configuration model (represented as CSP). An abbreviation is used for the constraint representation; for example, CA ↔ MP is the short form of CA = 1 ↔ MP = 1. MP = Mobile Phone, CA = Calls, GPS = GPS, SC = Screen, CM = Camera, MPX = MPX Player, BA = Basic, CO = Color, HR = High Resolution.

CSP Solution Search: Forward Checking

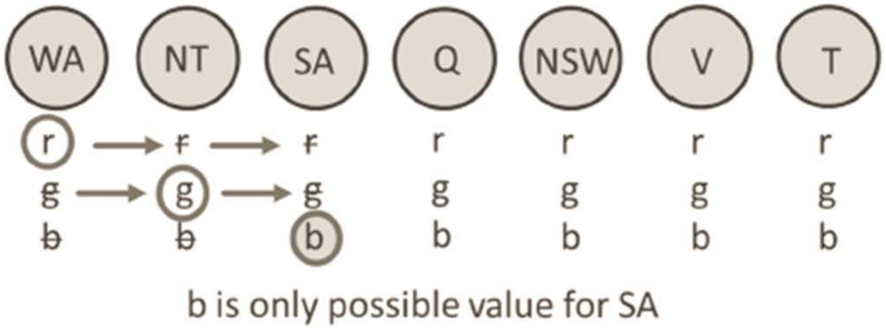


FIGURE 6.4

A simple example of forward checking. The variables WA and NT already have assigned values (*r* and *g*). The only possible remaining value for SA is *b*; *r* and *g* do not have to be checked for consistency with the settings of WA and NT.

Dynamic Constraint Satisfaction

- Reasoning over variables states
- Only active variables are part of the solution
- Activation constraints determine activity status of a variable
- HighResolution (Camera) = yes → active(HighResolution).

[Mittal and Falkenhainer, 1990]

Generative Constraint Satisfaction (GCSP)

- Representational limits of discussed approaches
- Component-oriented representation not possible (only variables and constraints)
- Not applicable if number of components depends on the preferences of a user
- Need for „on the fly“ generation of components

Generative Constraint Satisfaction (GCSP)

PC P DESCRIPTION:

P.name := [String];

P.price := [Integer];

P.usage := {'internet','scientific','multimedia'};

P.efficiency := {'A','B','C'};

P.PORTS := {screen-of-pc-1[Screen], screen-of-pc-2[Screen],
hdunit-of-pc-1[HDUnit], ...};

P.screens := <screen-of-pc-1,screen-of-pc-2>;

P.mb := <mb-of-pc-1>;

P.hdunits := <hdunit-of-pc-1,hdunit-of-pc-2>;

P.efficiency = P.mb.efficiency; /* Example constraint*/

Solution Search (GCSP)

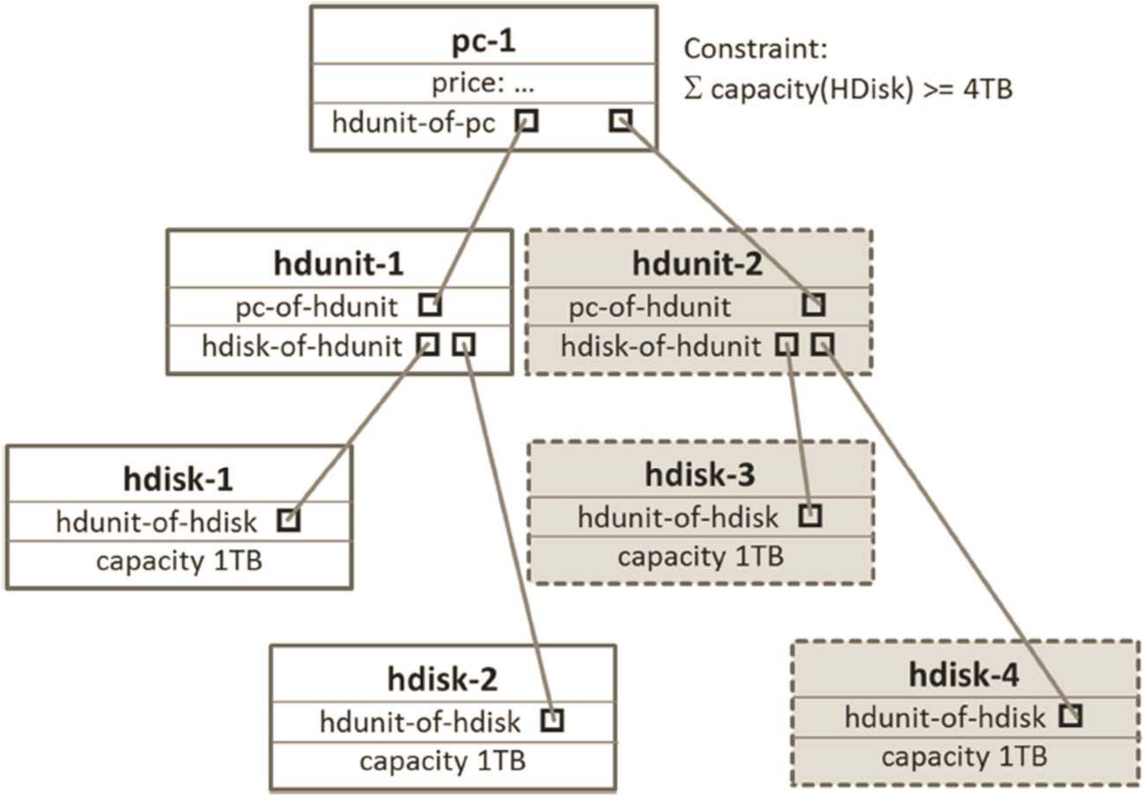


FIGURE 6.5

Simple GCSP dynamic component creation example. The constraint representation is simplified (i.e., it does not directly correspond to the GCSP constraint representation used in Stumptner et al., 1998).

Graphical Knowledge Representations

- Need to improve maintainability of configuration models
- Approach: graphical knowledge representations
- Automated translation into executable representation
- Examples:
  - Feature Models
  - UML Configuration Models

Feature Models

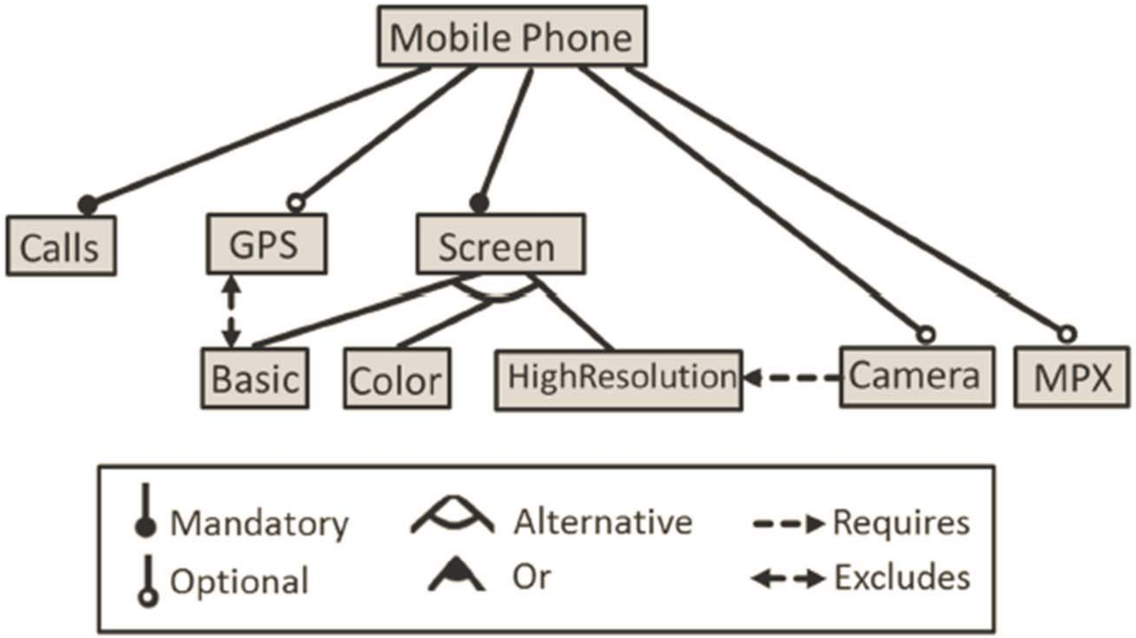


FIGURE 6.6

Feature model of a mobile phone (adapted version of Benavides et al., 2010). This feature model is equivalent to the constraint-based representation of Figure 6.3.

Semantics of Feature Models

Table 6.1 Semantics of feature models defined in terms of constraints (propositional logic). Features are represented by Boolean variables.

Relationship/Constraint	Semantics
mandatory(P,C)	$P \leftrightarrow C$
optional(P,C)	$C \rightarrow P$
or $(P, C_1, C_2, \ldots, C_n)$	$P \leftrightarrow (C_1 \vee C_2 \vee \ldots \vee C_n)$
alternative $(P, C_1, C_2, \ldots, C_n)$	$(C_1 \leftrightarrow (\neg C_2 \wedge \ldots \wedge \neg C_n \wedge P)) \wedge (C_2 \leftrightarrow (\neg C_1 \wedge \neg C_3 \wedge \ldots \wedge \neg C_n \wedge P))$ $\wedge \ldots \wedge (C_n \leftrightarrow (\neg C_1 \wedge \ldots \wedge \neg C_{n-1} \wedge P))$
requires(P,C)	$P \rightarrow C$
excludes(P,C)	$\neg P \vee \neg C$

UML Configuration Model

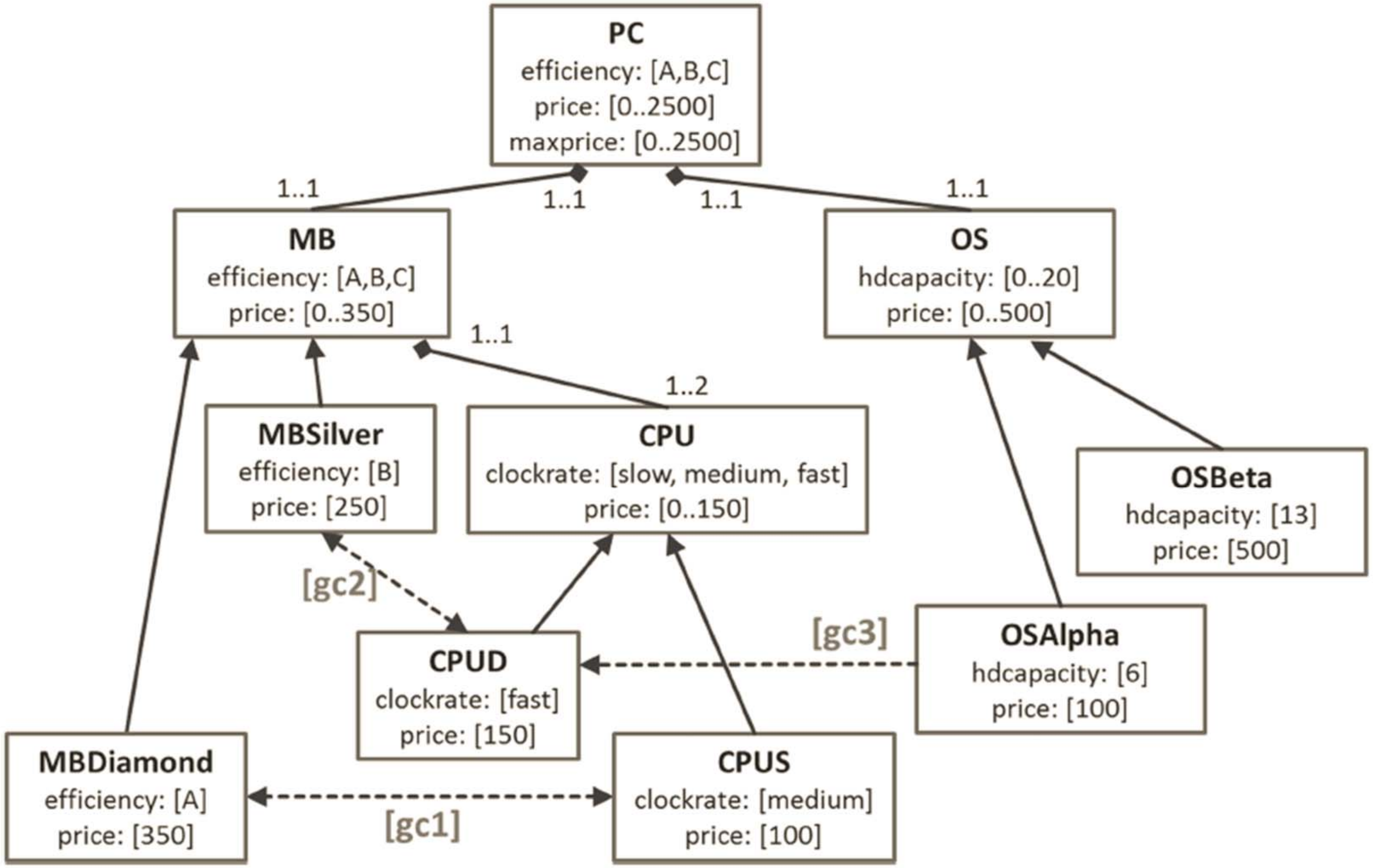


FIGURE 6.9

Fragment of the PC model (adapted part of Figure 6.7).

UML Configuration Model: Constraints

Table 6.3 Constraints related to the configuration model in Figure 6.9.

Name	Description
gc1	CPUs of type CPUS are incompatible with motherboards of type MBDiamond
gc2	CPUs of type CPUD are incompatible with motherboards of type MBSilver
gc3	Each OS of type OSAlpha requires a CPU of type CPUD
prc2′	The price of one personal computer (PC) is determined by the prices of the motherboard (MB), the CPUs, and the operating system (OS)
resc1	The computer price must be less or equal to the maxprice defined by the customer

UML Configuration Model: Formalization of Product Structure

Table 6.4 Example formalizations of the model (C_{KB}) depicted in Figure 6.9. *getcpus* denotes a collection operator (Felfernig et al., 2000a) that collects all *cpus* connected with motherboard Y. For reasons of readability we limit the example to attribute range restrictions (e.g., PC(efficiency)).

Modeling Facility	Example in FOL
Component types	{PC/1, MB/1, MBDiamond/1, MBSilver/1, CPU/1, CPUS/1, CPUD/1, OS/1, OSAlpha/1, OSBeta/1} $\subseteq$ *CLANG*
Attributes	{efficiency/2, price/2, maxprice/2, clockrate/2, hdcapacity/2} $\subseteq$ *CLANG*
Relationships	{pc-of-mb/2, mb-of-pc/2, mb-of-cpu/2, cpu-of-mb/2, pc-of-os/2, os-of-pc/2} $\subseteq$ *CLANG*
PC (efficiency)	$\{\forall X : PC(X) \rightarrow \exists_1^1 A_X : efficiency(X, A_X) \wedge A_X = A \vee A_X = B \vee A_X = C.\} \subseteq C_{KB}$
MB (efficiency)	$\{\forall X : MB(X) \rightarrow \exists_1^1 A_X : efficiency(X, A_X) \wedge A_X = A \vee A_X = B \vee A_X = C.\} \subseteq C_{KB}$
MB (price)	$\{\forall X : MB(X) \rightarrow \exists_1^1 A_X : price(X, A_X) \wedge A_X \geq 0 \wedge A_X \leq 350.\} \subseteq C_{KB}$
CPUS (price)	$\{\forall X : CPUS(X) \rightarrow \exists_1^1 A_X : price(X, A_X) \wedge A_X = 100.\} \subseteq C_{KB}$
part-of (PC,MB)	$\{\forall X : PC(X) \rightarrow \exists_1^1 Y : MB(Y) \wedge \text{pc-of-mb}(X, Y).\} \subseteq C_{KB}$ $\{\forall X : MB(X) \rightarrow \exists_1^1 Y : PC(Y) \wedge \text{mb-of-pc}(X, Y).\} \subseteq C_{KB}$
part-of (PC,OS)	$\{\forall X : PC(X) \rightarrow \exists_1^1 Y : OS(Y) \wedge \text{pc-of-os}(X, Y)\} \subseteq C_{KB}$ $\{\forall X : OS(X) \rightarrow \exists_1^1 Y : PC(Y) \wedge \text{os-of-pc}(X, Y).\} \subseteq C_{KB}$

UML Configuration Model: Formalization of Constraints

gc1	$\{\forall X, Y: \text{mb-of-cpu}(X, Y) \wedge MBDiamond(X) \wedge CPUS(Y) \rightarrow false.\} \subseteq C_{KB}$
gc2	$\{\forall X, Y: \text{mb-of-cpu}(X, Y) \wedge MBSilver(X) \wedge CPUD(Y) \rightarrow false.\} \subseteq C_{KB}$
gc3	$\{\forall X, Y: PC(X) \wedge OSAlpha(Y) \wedge$ $\text{pc-of-os}(X, Y) \rightarrow \exists_1^1 U, V: MB(U) \wedge CPUD(V) \wedge \text{pc-of-mb}(X, U) \wedge$ $\text{mb-of-cpu}(U, V).\} \subseteq C_{KB}$
prc2'	$\{\forall X: PC(X) \wedge price(X, PCP) \wedge \text{pc-of-mb}(X, Y) \wedge$ $\text{pc-of-os}(X, Z) \wedge getcpus(Y, CPUs) \rightarrow PCP =$ $\sum_{c \in \{Y,Z\} \cup CPUs \wedge price(c,P)} P.\} \subseteq C_{KB}$
resc1	$\{\forall X: PC(X) \wedge price(X, PCP) \wedge maxprice(X, PCMP) \rightarrow PCP \leq PCMP.\} \subseteq$ C_{KB}

Exercises

1. Explain the concept of forward checking on the basis of an example.

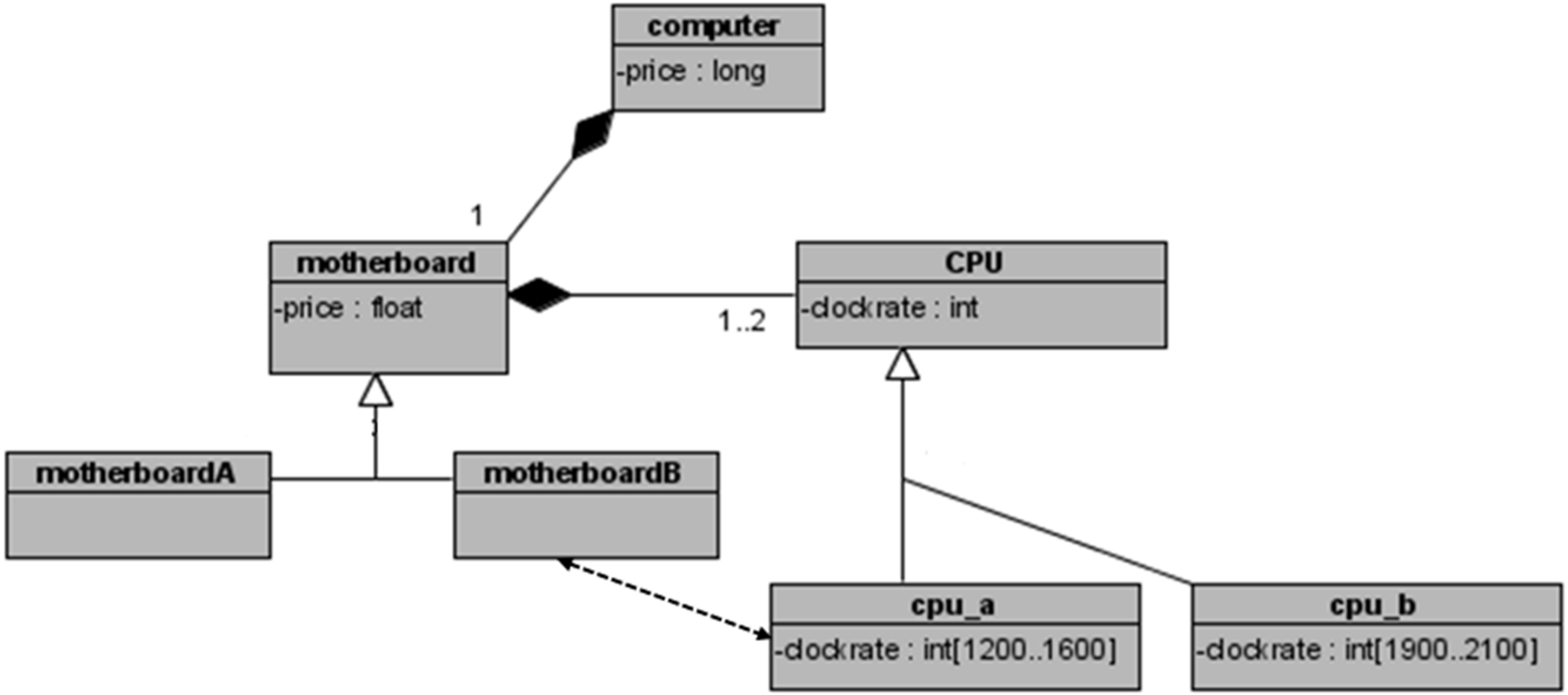
2. Translate the Mobile Phone feature model into a corresponding CSP-based representation.

3. Implement the Mobile Phone feature model with the CHOCO constraint solver (http://choco-solver.org)

4. Develop a feature model for a product domain of your own choice (not discussed in lecture).

5. Translate the following UML Model (next slide) into a logic-based representation.

Exercises (UML Model)



Thank You!

References (1)

(1) Amilhastre, J., Fargier, H., Marquis, P., 2002. Consistency restoration and explanations in dynamic CSPs - application to configuration. Artificial Intelligence 135 (1–2), 199–234.

(2) Andersen, H., Hadzic, T., Pisinger, D., 2010. Interactive cost configuration over decision diagrams. Journal of Artificial Intelligence Research 37, 99–139.

(3) Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (Eds.), 2003. The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, New York, NY.

(4) Bacchus, F., vanBeek, P., 1998. On the conversion between non-binary and binary constraint satisfaction problems. In: 15th National Conference on Artificial Intelligence (AAAI'98). Madison, Wisconsin, pp. 311–318.

(5) Batory, D., 2005. Feature models, grammars, and propositional formulas. In: Obbink, H., Pohl, K. (Eds.), Software Product Lines Conference. Lecture Notes in Computer Science, vol. 3714. Springer, Rennes, France, pp. 7–20.

(6) Benavides, D., Segura, S., Ruiz-Cortés, A., 2010. Automated analysis of feature models 20 years later: a literature review. Information Systems 35 (6), 615–636.

References (2)

(7) Booch, G., Rumbaugh, J., Jacobson, I., 2005. Unified Modeling Language User Guide, second ed. Addison-Wesley, Reading, MA.

(8) Bowen, J., Bahler, D., 1991. Conditional existence of variables in generalized constraint networks. In: 9th National Conference on Artificial Intelligence (AAAI-91). Anaheim, California, pp. 215–220.

(9) Brailsford, S., Potts, C., Smith, B., 1999. Constraint satisfaction problems: algorithms and applications. European Journal of Operational Research 199 (3), 557–581.

(10)Brewka, G., Eiter, T., Truszczy´nski, M., 2011. Answer set programming at a glance. Communications of the ACM 54 (12), 92–103.

(11)Buchheit, M., Klein, R., Nutt,W., 1995. Constructive Problem Solving: A Model Construction Approach Towards Configuration, Deutsches Forschungszentrum für Künstliche Intelligenz, Saarbrücken (TM-95-01).

(12)Claessen, K., Een, N., Sheeran, M., Sörensson, N., 2008. SAT-solving in practice. In: 9th International Workshop on Discrete Event Systems. Göteborg, Sweden, pp. 61–67.

References (3)

(13)Czarnecki, K., Helsen, S., Eisenecker, U., 2005. Formalizing cardinality-based feature models and their specialization. SoftwareProcess: Improvement and Practice 10 (1), 7–29.

(14)Falkner, A., Haselböck, A., 2013. Challenges of knowledge evolution in practice. AI Communications 26 (1), 3–14.

(15)Falkner, A., Schreiner, H., 2014. SIEMENS: configuration and reconfiguration in industry. In: Felfernig, A., Hotz, L.,Bagley, C., Tiihonen, J. (Eds.),Knowledge-based Configuration – FromResearch to Business Cases. Morgan Kaufmann Publishers, Waltham, MA, pp. 199–210 (Chapter 16).

(16)Falkner, A., Felfernig, A., Haag, A., 2011. Recommendation technologies for configurable products. AI Magazine 32 (3), 99–108.

(17)Faltings, B., Weigel, R., 1994. Constraint-based knowledge representation for configuration systems. Tech. Rep. TR-94/59, Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland.

(18)Felfernig, A., 2007. Standardized configuration knowledge representations as technological foundation for mass customization. IEEE Transactions on Engineering Management 54 (1), 41–56.

References (4)

(19)Felfernig, A., Burke, R., 2008. Constraint-based recommender systems: technologies and research issues. In: ACM International Conference on Electronic Commerce (ICEC08). Innsbruck, Austria, pp. 17–26.

(20)Felfernig, A., Zanker, M., 2000. Diagrammatic acquisition of functional knowledge for product configuration systems with the unified modeling language. In: Proceedings of the First International Conference on Theory and Application of Diagrams (Diagrams ’00). Lecture Notes in Computer Science, vol. 1889. Springer-Verlag, London, UK, pp. 361–375.

(21)Felfernig, A., Friedrich, G.E., Jannach, D., 2000a. UML as domain specific language for the construction of knowledge-based configuration systems. International Journal of Software Engineering and Knowledge Engineering 10 (4), 449–469.

(22)Felfernig, A., Jannach, D., Zanker, M., 2000b. Contextual diagrams as structuring mechanisms for designing configuration knowledge bases in UML. In: 3rd International Conference on the Unified Modeling Language (UML2000). Lecture Notes in Computer Science, Vol. 1939. Springer, York, UK, pp. 240–254.

References (5)

(23)Felfernig, A., Friedrich, G., Jannach, D., 2001. Conceptual modeling for configuration of mass-customizable products. Artificial Intelligence in Engineering 15 (2), 165–176.

(24)Felfernig, A., Friedrich,G., Jannach,D., Stumptner, M.,Zanker, M., 2003. Configuration knowledge representations for semantic web applications. Artificial Intelligence for Engineering Design, Analysis andManufacturing (AIEDAM) 17 (1), 31–50.

(25)Felfernig, A., Friedrich,G., Jannach,D., Stumptner, M., 2004. Consistency-based diagnosis of configuration knowledge bases. Artificial Intelligence 152 (2), 213–234.

(26)Felfernig, A., Schubert, M., Zehentner, C., 2012. An efficient diagnosis algorithm for inconsistent constraint sets. Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AI EDAM) 26 (1), 53–62.

(27)Felfernig, A., Benavides, D., Galindo, J., Reinfrank, F., 2013a. Towards anomaly explanation in feature models. In: Workshop on Configuration. Vienna, Austria, pp. 117–124.

References (6)

(28)Felfernig, A., Reiterer, S., Stettinger, M., Reinfrank, F., Jeran, M., Ninaus, G., 2013b. Recommender systems for configuration knowledge engineering. In: Workshop on Configuration. Vienna, Austria, pp. 51–54.

(29)Fleischanderl, G., Friedrich, G.E., Haselböck, A., Schreiner, H., Stumptner, M., 1998. Configuring large systems using generative constraint satisfaction. IEEE Intelligent Systems 13 (4), 59–68.

(30)Freuder, E., 1997. In pursuit of the holy grail. Constraints 2 (1), 57–61.

(31)Friedrich, G., Jannach, D., Stumptner, M., Zanker, M., 2014. Knowledge engineering for configuration systems. In: Felfernig, A., Hotz, L., Bagley, C., Tiihonen, J. (Eds.), Knowledge-based Configuration – From Research to Business Cases. Morgan Kaufmann Publishers, Waltham, MA, pp. 139–155 (Chapter 11).

(32)Gebser, M., Kaminski, R., Ostrowski, M., Schaub, T., Thiele, S., 2009. On the input language of ASP grounder Gringo. In: Erdem, E., Lin, F., Schaub, T. (Eds.), Proceedings of the 10th International Conference on Logic Programming and Nonmonotonic Reasoning. Lecture Notes in Computer Science, Vol. 5373. Springer, Potsdam, Germany, pp. 502–508.

References (7)

(33)Gelfond, M., Lifschitz, V., 1988. The stable model semantics for logic programming. In: Kowalkowski, R.A., Bowen, K.A. (Eds.), Logic Programming: Proceedings of the 5th International Conference and Symposium. Seattle, WA, pp. 1070–1080.

(34)Giunchiglia, E., Lierler, Y., Maratea, M., 2006. Answer set programming based on propositional satisfiability. Journal of Automated Reasoning 36 (4), 345–377.

(35)Gomes, C., Kautz, H., Sabharwal, A., Selman, B., 2008. Satisfiability solvers. In: Handbook of Knowledge Representation. Foundations of Artificial Intelligence, vol. 3. Elsevier, New York, NY, pp. 89–134 (Chapter 2).

(36)Günter, A., Kühn, C., 1999. Knowledge-Based Configuration - Survey and Future Directions. In: Puppe, F. (Ed.), XPS-99: knowledge based systems, proceedings 5th biannual german conference on knowledge based systems. Lecture Notes in Artificial Intelligence, Vol. 1570. Springer, Würzburg.

(37)Hotz, L., Günter, A., 2014. KONWERK. In: Felfernig, A., Hotz, L., Bagley, C., Tiihonen, J. (Eds.), Knowledge-based Configuration – From Research to Business Cases. Morgan Kaufmann Publishers, Waltham, MA, pp. 281–295 (Chapter 24).

References (8)

(38)Hotz, L., Neumann, B., 2005. Scene Interpretation as a Configuration Task. Künstliche Intelligenz 3, 59–65.

(39)Hotz, L., Wolter, K., 2014. Smarthome Configuration Model. In: Felfernig, A., Hotz, L., Bagley, C., Tiihonen, J. (Eds.), Knowledge-based Configuration – From Research to Business Cases. Morgan Kaufmann Publishers, Waltham, MA, pp. 121–135 (Chapter 10).

(40)Hotz, L., Wolter, K., Krebs, T., Deelstra, S., Sinnema, M., Nijhuis, J., MacGregor, J., 2006. Configuration in Industrial Product Families - The ConIPF Methodology. IOS Press, Berlin.

(41)Jannach, D., Zanker, M., Felfernig, A., Friedrich, G., 2010. Recommender Systems: An Introduction. Cambridge University Press, MA.

(42)Janota, M., 2008. Do SAT solvers make good configurators? In: First Workshop on Analyses of Software Product Lines (ASPL08) of 12th International Software Product Lines Conference (SPLC08). Limerick, Ireland, pp. 10–14.

References (9)

(43)Janota, M., Botterweck, G., Grigore, R., Marques-Silva, J., 2010. How to complete an interactive configuration process? In: 36th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2010). Lecture Notes in Computer Science, vol. 5901. Springer, Czech Republic, pp. 528–539.

(44) Jüngst, W., Heinrich, M., 1998. Using resource balancing to configure modular systems. IEEE Intelligent Systems 13 (4), 50–58.

(45)Junker, U., 2004. QUICKXPLAIN: preferred explanations and relaxations for over-constrained problems. In: McGuinness, D.L., Ferguson, G. (Eds.), 19th Intl. Conference on Artifical Intelligence (AAAI'04). AAAI Press, pp. 167–172.

(46)Junker, U., 2006. Configuration. In: Rossi, F., vanBeek, P.,Walsh, T. (Eds.), Handbook of Constraint Programming. Foundations of Artificial Intelligence, vol. 2. Elsevier, New York, NY, pp. 837–873.

(47)Junker, U., Mailharro, D., 2003. The logic of ILOG (J)Configurator: Combining constraint programming with a description logic. In: 18th International Joint Conference on Artificial Intelligence (IJCAI-03), Configuration Workshop. Acapulco, Mexico, pp. 13–20.

References (10)

(48)Kang, K., Cohen, S., Hess, J., Novak, W., Peterson, S., 1990. Feature-Oriented Domain Analysis Feasibility Study (FODA). Tech. Rep. CMU/SEI-90-TR-021, Carnegie Mellon University, Software Engineering Institute, Pittsburgh.

(49)Kiziltan, Z., Hnich, B., 2001. Symmetry breaking in a rack configuration problem. In: Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001), Workshop on Modelling and Solving Problems with Constraints. Seattle, WA, pp. 1–6.

(50)Leitner,G., Felfernig, A.,Blazek, P.,Reinfrank, F.,Ninaus, G., 2014. User interfaces for configuration environments. In: Felfernig, A., Hotz, L., Bagley, C., Tiihonen, J. (Eds.), Knowledge-based Configuration – From Research to Business Cases. Morgan Kaufmann Publishers, Waltham, MA, pp. 89–106 (Chapter 8).

(51)Li, B., Chen, L., Huang, Z., Zhong, Y., 2006. Product configuration optimization using a multiobjective genetic algorithm. International Journal of Advanced Manufacturing Technology 30 (1–2), 20–29.

(52)Mackworth, A., 1977. Consistency in networks of relations. Artificial Intelligence 8 (1), 99–118.

References (11)

(53)Mailharro, D., 1998. A classification and constraint-based framework for configuration. Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AI EDAM) 12 (4), 383–397.

(54)McDermott, J., 1982. R1: a rule-based configurer of computer systems. Artificial Intelligence 19 (1), 39–88.

(55)McGuinness, D., Wright, J., 1998. An industrial-strength description logic-based configurator platform. Intelligent Systems and their Applications, IEEE 13 (4), 69–77.

(56)Mittal, S., Falkenhainer, B., 1990. Dynamic constraint satisfaction problems. In: Proceedings of the eighth national conference on artificial intelligence (AAAI-90). Boston, MA, pp. 25–32 Aug.

(57)Mittal, S., Frayman, F., 1989. Towards a generic model of configuration tasks. 11th International Joint Conference on Artificial Intelligence (IJCAI-89). Detroit, Michigan, vol. 2 pp. 1395–1401.

(58)Neumann, B., 1988. Configuration expert systems: a case study and tutorial. In: Bunke, H.O. (Ed.), Proc. 1988 SGAICO Conference on Artificial Intelligence in Manufacturing, Assembly, and Robotics. Oldenbourg, Munich, pp. 27–68.

References (12)

(59)Pinedo, M., 2012. Scheduling: Theory, Algorithms, and Systems, fourth ed. Springer, New York, NY.

(60)Pitiot, P.,Aldanondo, M.,Vareilles, E.,Gaborit, P., Djefel, M.S.C., 2013. Concurrent product configuration and process planning, towards an approach combining interactivity and optimality. International Journal of Production Research 51 (2), 524–541.

(61)Puget, J.-F., Albert, P., 1991. PECOS: programmation par contraintes orientée objets. Génie logiciel et systèmes experts (GLSE) 23, 100–105.

(62)Reiter, R., 1987. A theory of diagnosis from first principles. Artificial Intelligence 32 (1), 57–95.

(63)Rossi, F., vanBeek, P., Walsh, T. (Eds.), 2006. Handbook of Constraint Programming. Foundations of Artificial Intelligence. Elsevier, New York, NY.

(64)Russel, S., Norvig, P., 2003. Artificial Intelligence – A Modern Approach, 2nd Edition. Prentice Hall, Upper Saddle River, NJ.

(65)Sabin, D.,Weigel, R., 1998. Product configuration frameworks – a survey. IEEE Intelligent Systems 13 (4), 42–49.

References (13)

(66)Schröder, C., Möller, R., Lutz, C., 1996. A partial logical reconstruction of PLAKON/KONWERK. In: Baader, F., Bürckert, H.-J., Günther, A., Nutt, W. (Eds.), Proceedings of theWorkshop on Knowledge Representation and Configuration WRKP'96. No. D-96-04 in DFKI Documents, pp. 55–64.

(67)Sinz,C., Haag,A., Narodytska, N.,Walsh,T., Gelle,E., Sabin, M., Junker,U., O'Sullivan, B.,Rabiser,R., Dhungana, D., Grünbacher, P., Lehner, K., Federspiel, C., Naus, D., 2007. Configuration. IEEE Intelligent Systems 22 (1), 78–90.

(68)Soininen, T., Tiihonen, J., Männistö, T., Sulonen, R., 1998. Towards a general ontology of configuration. Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AI EDAM) 12 (4), 357–372.

(69)Soininen, T., Niemelä, I., Tiihonen, J., Sulonen, R., 2001. Representing configuration knowledge with weight constraint rules. In: Provetti, A., Son, T.C. (Eds.), 1st International Workshop on Answer Set Programming: Towards Efficient and Scalable Knowledge (AAAI Technical, Report SS-01-01). pp. 195–201.

References (14)

(70)Soloway, E., Bachant, J., Jensen, K., 1987. Assessing the maintainability of XCON-in-RIME: coping with the problem of very large rule-bases. In: Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87). Seattle, Washington, pp. 824–829 July 13–17.

(71)Stumptner, M., 1997. An overview of knowledge-based configuration. AI Communications 10 (2), 111–126.

(72)Stumptner, M., Friedrich, G., Haselböck, A., 1998. Generative constraint-based configuration of large technical systems. Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AI EDAM) 12 (4), 307–320.

(73)Terashima-Marin, H., Ross, P.,Valenzuela-Rendon, M., 2008. Hyper-heuristics for the dynamic variable ordering in constraint satisfaction problems. In: Genetic and Evolutionary Computation Conference (GECCO'08). Atlanta, Georgia, pp. 571–578.

(74)Tiihonen, J., Soininen, T., Niemelä, I., Sulonen, R., 2003. A practical tool for mass-customising configurable products. In: Proceedings of the 14th International Conference on Engineering Design, Stockholm, Sweden, August 19–21, 2003, CDROM, p. 10 (Paper number: 1290).

References (15)

(75) Tiihonen, J., Heiskala, M., Anderson, A., Soininen, T., 2013. WeCoTin – A practical logic-based sales configurator. AI Communications 26 (1), 99–131.

(76) Tsang, E., 1993. Foundations of Constraint Satisfaction. Academic Press, London, San Diego, New York.

(77) Warmer, J., Kleppe, A., 2003. The Object Constraint Language: Getting Your Models Ready for MDA, second ed. Addison-Wesley Longman Publishing, Boston, MA.