Cognition, Sociability, and Constraints*

Gerhard Weiß

Institut für Informatik, Technische Universität München
80290 München, Germany, weissg@in.tum.de

Abstract. This paper focuses on the challenge of building technical agents that act flexibly in modern computing and information environments. Flexibility is approached in terms of both cognition (ranging from reactive to pro-active) and sociability (ranging from isolated to interactive). It is argued that existing agent architectures tend to inherently limit an agent's flexibility because they imply a discrete cognitive and social behavior space. A generic constraint-centered architectural framework is proposed that aims at enabling agents to act in a continuous behavior space. According to this framework flexibility is not considered to be a property that can be directly implemented, but as a property that emerges as a result of an agent's continuous attempt to handle local and global constraints. The long-term challenge rised by the thoughts and ideas in this paper is to integrate various existing constraint-handling approaches and to valididate the usefulness of the generic framework through its implementation and application.

1 Introduction

Modern computing platforms and information environments are becoming more and more distributed, large, open, dynamic, and heterogeneous. Computers are no longer stand-alone systems, but have become tightly connected both with each other and their users. The increasing technological complexity of such platforms and environments goes together with an increasing complexity of their potential applications. This development has led to a rapidly growing research and application interest in agents as a powerful concept and guideline for designing, implementing, and analyzing complex information processing systems in which decentralized data are processed in an asynchronous but concerted way by several computers that act more as "individuals" rather than just "parts". There is no universally accepted definition of the term agent (see e.g. [16, 63] for more detailed considerations on this term), but generally it is used to refer to a *computational autonomous entity*, that is, an entity that runs on a computing device and to some extent decides upon the activities it takes in order to meet his design objectives independent of humans and other agents. Building agents that deserve to be called flexible is one of the major driving forces in the field of agent-based computing, where flexibility is usually assumed to be composed of the following three basic ingredients [30]:

* This is a shortened and revised version of [61].

- *reactivity*: the ability to respond in a timely fashion to environmental changes;
- *pro-activeness*: the ability to generate and rationally pursue goals; and
- *sociability*: the ability to interact with other agents and possibly humans by exchanging information, coordinating activities, and so forth.

In view of the requirements and challenges that arise in the context of modern computing systems and their applications, it is obvious that flexibility is a key property that agents need to possess in order to operate successfully. This paper takes a closer look on how to build agents that are flexible with respect to their cognitive (reactive and pro-active) *and* their social (isolated and interactive) behavior. The structure of the paper is as follows. Section 2 argues that existing agent architectures tend to inherently result in a limited flexibility because they imply a discrete behavior space. Section 3 proposes an alternative, constraint-centered view of achieving flexibility that aims at avoiding this limitation. Section 4 describes a generic architectural framework called CCAF that integrates the main thoughts and ideas of this alternative view. Section 5 concludes the paper with a summary, a note on related work, and an overview of open research issues. A number of pointers to related research efforts are provided throughout the paper.

2 Discrete versus Continuous Behavior Spaces

A wide variety of architectures—particular methodologies (including arrangements of data, algorithms, and control and data flows) for building agents—has been described in the literature (see, e.g., [24, 38, 62] for an overview). A major characteristic of existing architectures is that they are implicitly based on the assumption that reactivity, pro-activeness and inter-activeness are "pure", disjoint behavioral modes: there is nothing in between them, and every goal or task (or every context in which a goal or task is pursued) allows to uniquely and correctly determine at every time which of the "pure modes" is appropriate. This assumption is very critical and inherently restricts an agent's flexibility, because the "pure modes" define a discrete behavior space composed of just a few, disconnected behavioral "isles" or repertoires. Figure 1, which is based on the standard view of reactivity and pro-activeness as cognitive characterizations and of interactivity (as opposed to isolatedness) as a social characterization of an agent's behavior, illustrates this drawback.[1] Obviously, the repertoires allow only a very limited form of "graceful degradation" in an agent's behavior. Most importantly, "discrete behavior space architectures" require a mapping of *all* potential environmental situations and activity contexts to a few discrete repertoires at design time. Moreover, they require the identification of the criteria for switching between the pure modes, which is to say that they require to foresee all conditions and circumstances under which an agent will have to act reactively, pro-actively, and interactively in order to fulfill his mission.

[1] This figure takes care of the fact that many existing architectures do not capture what could be called "reactive interactivity" or "interactive reactivity"; instead, reactivity is usually associated with isolated behavior.

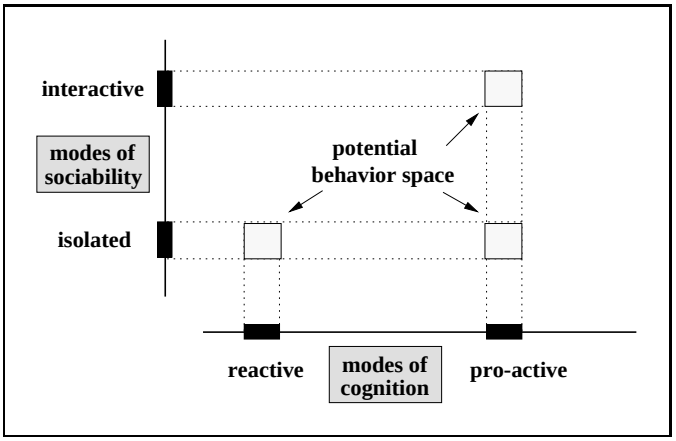

Fig. 1. The discrete behavior space metaphor.

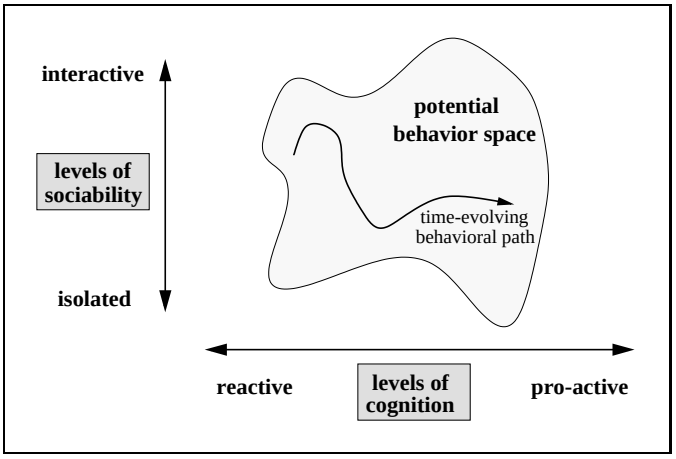

Fig. 2. The continuous behavior space metaphor.

In order to avoid the flexibility restrictions implied by existing architectures, an extended and more natural view of an agent's behavior space is required. An obvious alternative would be to not assume the existence of just two social and two cognitive discrete modes, but to have continuous levels of sociability and cognition. According to this view, an agent is not flexible in that he switches between a few behavioral repertoires, but is flexible at various levels: sociability now varies from isolated to interactive, and cognition now varies from reactive to pro-active as a continuum. Figure 2 illustrates this view. In the next section it is argued that this alternative can be realized under certain conditions through constraint handling.

3 Toward a Constraint-centered Realization

From what has been said above it follows that some sort of behavior-influencing mechanism is needed that enables an agent to act in a continuous behavior space on multiple levels of sociability and cognition. An intuitively obvious and appealing candidate for such a mechanism is constraint handling. It is an obvious candidate because agents can be considered, in a very natural way, as entities that are surrounded by and must cope with various constraints that do have, or should have, a significant impact on both their social and cognitive behavior. The most common examples of constraints are time bounds, cost bounds, and requirements on the desired solution quality. These three constraints are present in one or another form in most, if not all, application domains. Other examples of constraints that are available at the agent and group levels are an agent's individual preferences, collective preferences of groups, psychological or social commitments, limitations on resources an agent or a group wants to use, roles an agent is expected to play in joint processes, norms an agent has to take into consideration, conventions that apply in an agent's application domain, and so forth. It is an appealing candidate because if offers a new perspective of isolated activity, interactivity, reactivity, and pro-activeness as observable properties that *emerge* as a result of an agent's continuous attempt to handle local and global constraints while he tries to meet his design perspective. Constraint handling inherently offers the possibility, or even requires, to act at different levels of sociability and cognition, and an agent can be said to be cognitively and socially flexible to the degree he can handle—identify and (fully or partially) satisfy—the constraints he is confronted with. This is in clear contrast to the standard, discrete behavior space perspective that aims at achieving flexibility through directly implementing pure behavioral modes and appropriately switching between them. Having said these more general, motivational things, a closer look has to be taken on the specific conditions under which "continuous flexibility" can be achieved through constraint handling.

Condition 1. Most obviously, constraints on the one hand and an agent's internal activities (ranging from environmental analysis over planning and learning to communication) on the other must be tightly intertwined:

- an agent should be able to dynamically modify and refine the set of constraints as he runs any internal activity such that ideally each internal activity can result in a modified constraint set (*constraint identification and acquisition*); and
- an agent should be able to execute each of his internal activities under the given constraints such that ideally none of his activities violates constraints (*constraint satisfaction*).

Here are two simple illustrations of this intertwining:

- An agent analyzing the current sensor input finds out that the resources he would need for successfully pursuing some of his goals are limited or that

the access to these resources need to be coordinated with other agents. The limitation constraint may have a significant impact on the goal ranking, while the coordination constraint may require to initiate negotiation and contracting activities.
- New commitments, roles, and responsibilities may arise as a result of a negotiation or contracting process, and these new constraints may require an agent to revise his planned future activities.

These illustrations also show that an agent has to be able to cope with the fact that constraints can dynamically raise or disappear as a result of his own activities. Obviously, realizing this intertwining is particularly difficult in domains in which constraints change dynamically due to the activities of other agents' or humans. Related work dealing with dynamic constraint satisfaction is, e.g., [4, 9, 54].

Condition 2. An agent must be able to carry out each activity he considers as relevant for a solution process in cooperation with other agents, whenever the constraints he faces require this. (Here coordination means that an agent carries out an activity together with others or that an activity is completely delegated to one or several other agents. Note that available architectures typically restrict coordination to script generation/planning, whereas e.g. situation analysis and failure analysis are often treated as local activities that are not distributed.) The following basic examples illustrate this condition:

- An agent considers the possibility to share some time-critical control task with one or several other agents, if control appear to be too time-consuming for him. After having communicated with other agents (where this communication itself is subject to the time constraint), the agent comes to the conclusion that distribution of control would result in a faster solution and thus tries to establish "control contracts." The set of agents he considers as potential cooperators may be constrained by commitments in which he is already involved, and time pressure also guides him in his choice of the method he uses for selecting cooperators (e.g., iterated negotiation or voting).
- If an agent is required to provide a low-cost solution but communication with other agents is expensive due to limited available bandwidth, then he will avoid negotiation activities even if this results in a solution of low(er) quality.

As these examples show, taking the constraint-centered view serious means that communicative and coordinative acts among agents have to be designed, to a large extent, in a strictly constraint-triggered way. This also means that the contents of communicative acts are determined by constraints to a remarkable degree. Taking an extreme position, it can be even argued that constraints (and nothing else) constitute the "necessary and sufficient conditions" for communicative and coordinative acts—the agents communicate and coordinate *if and only if* this is required by the constraints they have to fulfill.

It is important to see that this condition does also concern an agent's constraint handling process itself: constraints can require an agent to identify and

satisfy constraints in cooperation with others. This can be the case, for instance, if constraint identification requires to search through databases which the agent simply is not allowed to access or if constraint satisfaction is too complex to be solvable by the agent without violating available time/cost constraints. Available work that is relevant w.r.t. the distribution of the constraint handling process is, e.g., [13, 26, 52, 53, 64, 65].

Condition 3. Agents must be able to reason about their constraints, and to involve other agents into this reasoning process whenever necessary. Following the argumentation in [33] (see also [10] for earlier considerations on this subject), this kind of reasoning must be quantitative in nature, because qualitative, purely symbolic reasoning about constraints like time and costs can be extremely complex especially in large-scale agent and multiagent contexts. More specifically, this means that achieving continuous flexibility through constraint handling requires an agent to be able

- to assign quantitative values to the constraints that express their relative importance they have for him;
- to assign quantitative values to the constraints that express the degrees to which he is willing to violate them;
- to assign quantitative values to the constraints that express the estimated risk of violating them (given the current environmental circumstances and the activity sequence he intends to execute); and
- to communicate (exchange, negotiate, refine, etc.) these quantities with other agents and humans.

This ability constitutes a profound basis for applying efficient mechanisms for (joint) constraint relaxation and propagation. In particular, based on these quantities an agent or a group of agents can efficiently evaluate the worth of alternative activities or activity sequences under the given constraints, supervise (joint) solution processes, and gracefully redirect solution processes in the case of constraint violation. Again here are some simple illustrations:

- An agent quantitatively expresses the degree to which he prefers a qualitatively high to a cheap solution in the case of his goal A, and how he weights the costs compared to the processing time in the case of his goal B.
- An agent quantitatively describes to what extent he is willing to relax role and normative constraints under the condition that this relaxation helps to cope with resource constraints.
- An agent decides to follow a cheaper "computation-sensitive" solution path rather than a more expensive "communication-sensitive" path. If it turns out during the solution process that the quality of the solution tends to be lower than he expected, then he reconsiders his choice and decide in a quantifiable way how to proceed from that point of the solution process such that the violation of the quality constraint is kept as minimal as possible.
- An agent asks other agents for analyzing the current situation and informs them about his "constraint preferences." The other agents make offers and quantitatively specify to what degree they could fulfill the constraints.

Related work dealing with the quantification of constraints in the context of scheduling is presented in [21, 20] (design-to-time scheduling) and [55, 57, 56, 59] (design-to-criteria scheduling). Related research on the quantification of motivations, which can be also viewed as constraints an agent has to satisfy, is reported in [58].

Condition 4. An agent must be able to efficiently handle constraints as he carries out any internal activity. This requires that an agent's reasoning about constraints (including key activities like constraint identification, relaxation, and quantification) is realized in a centralized way. In principle, it would be possible to treat constraint reasoning as a distributed, internal process. In this case, different internal activities (e.g., negotiation and exception handling) would independently of each other take care of available constraints. However, because in general *(i)* constraints are highly dynamic (they appear and disappear continuously as a result of an agent's own and other agents' activities), *(ii)* different internal activities are concerned by the same constraints, and *(iii)* constraint handling itself should not violate available constraints, it is most likely that a distributed internal constraint handling process is inefficient and results in a poor performance both at the agent and the group level.

It is noted that the above mentioned conditions are not disjoint, but related to each other. It is also stressed that this list of conditions is not claimed to be complete; instead, it is argued that these conditions are most elementary for achieving sophisticated flexibility through constraint handling.

4 An Architectural Framework based on Constraints

Figure 3 shows a generic architectural agent framework, called Constraint-Centered Architectural Framework (CCAF), that aims at capturing the constraint-centered view of flexibility and its conditions. According to this generic framework, an agent is composed of several kernel management modules (coordination, adaptation, script, execution, and constraints), where each module consists of several elementary activities or processes that are carried out by an agent in order to meet his design objectives. The activities within the management modules use and modify different data or information sets, as indicated by the data flow arrows. The figure also shows the basic flow of control between the internal activities, where a control link from a box A to a box B means that B (or its components) can be invoked or activated by A (or its components). The CCAF includes several standard data links. For instance, as indicated by the corresponding data flow arrows, the communication and interaction protocols mainly concern the coordination management, and the performance data are of particular relevance to the adaptation and execution management modules. Similarly, the CCAF includes several standard control links. For instance, the control link from the script management module to the execution management module indicates that script activities are usually followed by execution activities, and the control link from the execution management module to the adaptation module

indicates that an agent usually tries—on the basis of the monitored performance data—to improve his overall performance through failure analysis and learning. In addition to that, the CCAF integrates the "constraint-specific conditions" mentioned in the previous section as follows. Condition 4 ("centralized realization of an agent's constraint handling activities") requires the separate constraint management module in which all basic constraint handling activities run in a centralized manner. Condition 3 ("quantitative reasoning about constraints") requires the existence of explicit reasoning and quantification processes located within the constraint management module. Condition 1 ("tight intertwining between internal activities and constraints") results in the bidirectional data flow links between the set of constraints and each of the management modules. Condition 2 ("constraint-induced cooperation") implies the control links from all other modules to the coordination management module. Finally, the conditions 1 and 2 additionally require that there are control links from the constraint management module to all other management modules.

5 Discussion

Summary. The paper started from the observation that the concept of agents has the capacity to play an important role in understanding, managing and using distributed, large-scale, dynamic, open, and heterogeneous computing and information environments, and pointed out that flexibility (i.e., the ability to act reactively, pro-actively, and socially) is a desirable key property of agents. It then was argued that the flexibility that can be achieved through existing agent architectures is inherently limited, because they tend to generate a discrete behavior space that consists of just a few behavioral repertoires. An architectural framework, called CCAF, was introduced that aims at avoiding this limitation. The basic idea, or claim, behind the CCAF is that flexible behavior emerges along two continuous axis—cognition and sociability—as a result of an agent's attempt to handle constraints. Four conditions were identified that are considered to be essential for achieving continuous flexibility through constraint handling. With the help of these conditions the constraints inherently determine an agent's behavioral path by both preventing and enforcing activities that an agent carries out in order to achieve his goals. There are two contrary views of constraints: constraints as "a bad thing an agent has to cope with," and constraints as "a good thing that supports an agent in navigating through the space of possible cognitive and social behaviors." The CCAF, in some sense, brings these two perspectives together.

A Note on Related Work. Constraint handling is a well known topic in artificial intelligence in general (e.g., see [17, 35] and also the constraints archive at http://www.cs.unh.edu/ccc/archive/) and in the field of agent-based systems in particular (e.g., [14, 18]), and it is beyond the intention of this paper to comprehensively list all the works that are related to the thoughts and ideas presented here (various pointers have been already provided above and several others follow below). Two agent architectures that should be mentioned here because of

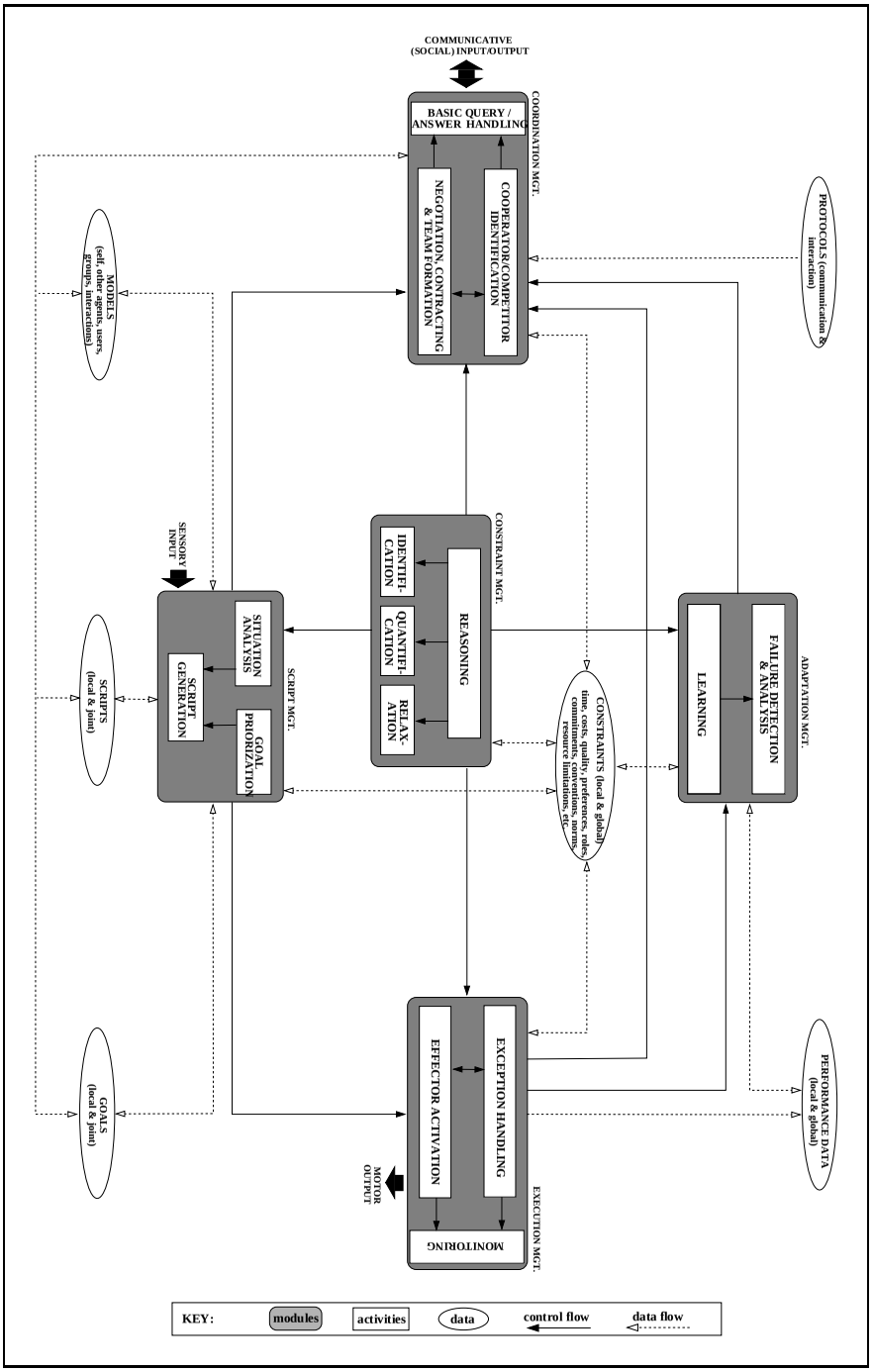

Fig. 3. The Constraint-Centered Architectural Framework (CCAF).

their close relationship to CCAF are EXCALIBUR (e.g., [40]) and WAFFLER (e.g., [1]). Both EXCALIBUR and WAFFLER focus on agents acting in real-time scenarios and both aim at exploiting constraint-oriented reasoning as a method for balancing reactivity and pro-activeness. For that reason both architectures can be viewed as instantiations of the CCAF. Compared to EXCALIBUR and WAFFLER the CCAF is broader in its scope and vision of the potential role of constraints and constraint handling for agents. In particular, CCAF does not only deal with the cognitive dimension ("reactivity versus pro-activeness") but also with the social dimension ("isolatedness versus interactivity") of flexibility from a unified constraint-centered perspective.

Open Issues. The constraint-oriented view of building flexible agents brings up several research issues, where the following appear to be particularly challenging:

- An agent's or a group's quantitative reasoning about constraints must be bounded, that is, like all other activities this reasoning has to be conducted under the available constraints. However, currently it is unknown how the problem of "constraint violation through constraint handling" can be solved efficiently. More generally, this requires to clarify how the CCAF and its "quantitative meta-reasoning about constraints" conception is related to the concepts of metalevel rationality (e.g., [43–45]) and social rationality [27] and to utilitarian—decision-theoretic—meta-reasoning (e.g., [7, 28]).
- There is a need for a "global constraint metrics" that allows multiple agents to unambiguously reason and communicate about their constraints and how to handle them. Such a metrics must be unique and known to all cooperating agents within a task domain, but may vary for different domains.
- As coordination and communication themselves are subject to the constraints, it has to be clarified whether this raises specific requirements on the coordination/communication languages and protocols used by the agents.
- It is broadly accepted that autonomy is a key feature of agents, but it is unclear how this feature on the one hand and autonomy on the other are related to each other. Intuitively, it appears that constraints require, or inherently imply, variability and adjustability in an agent's autonomy ("constraints constrain autonomy"). Details of this requirement—its formalization and practical realization—still need to be clarified. Work that may be of help in this respect can be found in [39].
- Another important open question is how the constraint-centered approach to agent design scales up to societies consisting of hundreds and thousands of agents. What happens if more than just a dozen agents try to simultaneously and dynamically handle their constraints? How stable and robust is such a system, and are there any criteria for ensuring global stability and robustness?
- The algorithms that realize an agent's internal activities need to be highly constraint-sensitive, that is, they should be designed such that an agent can act in accordance with available constraints and in response to changes in his constraint set at each point of the overall problem solving process in which he is engaged. In view of the three most common constraints—time, costs,

and quality—, this means that there is a strong need for "any-time, any-cost, and any-quality algorithms." The insight that such algorithms play an important role in building intelligent systems is not new in AI, and there is available work that is of particular relevance from the point of view of CCAF. For instance, there are approaches to reactive and anytime planning (e.g., [5, 6, 11, 12, 15, 22, 23, 41] and also [31, 36]), real-time search (e.g.,[25, 29, 32]); constraint-based negotiation (e.g., [42, 48, 51]); negotiation/interaction-based constraint satisfaction (e.g., [8, 34]); anytime constraint satisfaction (e.g., [37, 60, 66]); "soft contracting" that allows agents to break contracts and thus to act more flexibly e.g. in response to unexpected changes in their constraints (e.g., [2, 49, 50, 47]); coalition/team formation under time and cost constraints (e.g, [46]); deadline-based multiagent scheduling [19]; and real-time learning (e.g., [3]).

Disclaimer. This paper attempts to contribute to our understanding of what it means to build "really flexible agents." This paper should not be seen as describing finished work. Instead, the CCAF is best viewed as a starting point for further discussion and research on computational flexibility, and as a plea for combining and integrating the various related approaches mentioned throughout this paper into a unified, monolithic and coherent whole. It is a major long-term challenge for future work to identify the benefits and limitations of the CCAF through practical implementations and experimental analysis.

Acknowledgments. Many of the ideas described above were stimulated by the various research efforts referred to in this paper. Victor Lesser drawed my attention to the important role of an agent's ability to meta-reasoning about constraints, and one of the reviewers made me aware of the potential relationship between this kind of meta-reasoning and existing approaches to utilitarian meta-reasoning.

References

1. J.. Anderson. Waffler: A constraint-directed approach to intelligent agent design. In E.C. Freuder, editor, *Constraints & Agents. Papers from the AAAI Workshop (Technical Report WS-97-05)*, pages 70–75. AAAI Press, Menlo Park, CA, 1997.
2. M.R. Andersson and T.W. Sandholm. Leveled commitment contracts with myopic and strategic agents. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, pages 38–45, 1998.
3. A.G. Barto, S.J. Bradtke, and S.P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72:81–138, 1995.
4. C. Bessier. Arc-consistency in a dynamic constraint satisfaction problem. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, pages 221–226, 1991.
5. M. Boddy and T.L. Dean. An analysis of time-dependent planning. In *Proceedings of the 7th National Conference on Artificial Intelligence (AAAI-88)*, pages 49–54, 1988.

6. M. Boddy and T.L. Dean. Deliberation scheduling for problem solving in time-constrained environments. *Artificial Intelligence*, 67:245–285, 1994.
7. J.S. Breese and M.R. Fehling. Control of problem-solving: Principles and architecture. In R.D. Shachter, T. Levitt, L. Kanal, and J. Lemmer, editors, *Uncertainty in Artificial Intelligence 4*. Elsevier/North-Holland, Amsterdam, London, New York, 1990.
8. S.E. Conry, K. Kuwabara, V.R. Lesser, and R.A. Meyer. Multistage negotiation for distributed constraint satisfaction. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1462–1477, 1991.
9. R. Dechter and A. Dechter. Belief maintenance in dynamic constraint networks. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 37–42, 1988.
10. K.S. Decker and V.R. Lesser. Quantitative modeling of complex environments. *International Journal of Intelligent Systems in Accounting, Finance, and Management*, 2(4):215–234, 1993.
11. V. Decugis and J. Ferber. Action selection in an autonomous agent with a hierarchical distributed reactive planning architecture. In *Proceedings of the Second International Conference on Autonomous Agents (Agents'98)*, pages 354–361, 1998.
12. M. Drummond and J. Bresina. Anytime synthetic projection: Maximizing the probability of goal satisfaction. In *Proceedings of the Eight National Conference on Artificial Intelligence (AAAI-90)*, pages 138–144, 1990.
13. P.S. Eaton and E.C. Freuder. Agent cooperation can compensate for agent ignorance in constraint satisfaction. In M. Tambe and P. Gmytrasiewicz, editors, *Agent Modeling Papers from the 1996 AAAI Workshop (Technical Report WS-96-02)*, pages 24–29. AAAI Press, 1996.
14. P.S. Eaton, E.C. Freuder, and R.J. Wallace. Constraints and agents: Confronting ignorance. *AI Magazine*, 2:51–66, 1998.
15. R.J. Firby. Modularity issues in reactive planning. In *Proceedings of the Third International Conference on AI Planning Systems*, pages 78–85, 1996.
16. S. Franklin and A. Graesser. Is it an agent, or just a program?: A taxonomy for autonomous agents. In J.P. Müller, M.J. Wooldridge, and N.R. Jennings, editors, *Intelligent Agents III*, Lecture Notes in Artificial in Artificial Intelligence, Vol. 1193, pages 21–36. Springer-Verlag, Berlin et al., 1997.
17. E.C. Freuder and A.K. Mackworth. *Constraint-based reasoning*. The MIT Press, Cambridge, Mass., 1994.
18. E.C. Freuder (Chair). Constraints & agents. Papers from the AAAI workshop. Technical Report WS-97-05, AAAI Press, Menlo Park, CA, 1997.
19. S. Fujita and V.R. Lesser. Centralized task distribution in the presence of uncertainty and time deadlines. In *Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS-96)*, pages 87–94, 1996.
20. A. Garvey, K. Decker, and V.R. Lesser. A negotiation-based interface between a real-time scheduler and a decision-maker. In M. Klein and S. Lander, editors, *Models of Conflict Management in Cooperative Problem Solving. Papers of the AAAI Workshop (Technical Report WS-94-04)*. AAAI Press, Menlo Park, CA, 1994.
21. A. Garvey and V.R. Lesser. Design-to-time real-time scheduling. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(6):1491–1502, 1993.
22. E. Gat. Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, pages 802–815, 1992.

23. M.P. Georgeff and A.L. Lansky. Reactive reasoning and planning. In *Proceedings of the 6th National Conference on Artificial Intelligence (AAAI-87)*, pages 677–682, 1987.
24. A. Haddadi and K. Sundermeyer. Belief-desire-intention architectures. In G.M.P. O'Hare and N.R. Jennings, editors, *Foundations of Distributed Artificial Intelligence*, pages 169–186. Wiley, New York et al., 1996.
25. B. Hamidzadeh and S. Shekhar. Deadline compliance, predictability, and on-line optimization in real-time problem solving. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 220–226, 1995.
26. S. Haridi, P. Van Roy, and G. Smolka. An overview of the design of Distributed Oz. In *Proceedings of the Second International Symposium on Parallel Symbolic Computation (PASCO'97)*, pages 176–187, 1997.
27. L.M. Hogg and N.R. Jennings. Socially rational agents. In *Proceedings of the AAAI Fall Symposium on Socially Intelligent Agents*, pages 61–63, 1997.
28. E.J. Horvitz. Problem-solving design: reasoning about computational value, trade-offs, and resources. In *Proceedings of the Second Annual NASA Research Forum*, pages 26–43, 1987.
29. T. Ishida. Real-time search for autonomous agents and multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 1:139–167, 1998.
30. N.R. Jennings, K. Sycara, and M.J. Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems*, 1:7–38, 1998.
31. K. Kanazawa and T. Dean. A model for projection and action. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pages 985–990, 1989.
32. R.E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42(2-3):189–211, 1990.
33. V.R. Lesser. Reflections on the nature of multi-agent coordination and its implications for an agent architecture. *Autonomous Agents and Multi-Agent Systems*, 1:89–111, 1998.
34. J. Liu and K.P. Sycara. Exploiting problem structure for distributed constraint optimization. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 246–253, 1995.
35. A.K. Mackworth. Constraint satisfaction. In S.C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, pages 285–293. Wiley, New York, 1992.
36. P. Maes. Situated agents can have goals. *Robotics and Autonomous Systems*, 6:49–70, 1990.
37. P. Morris. The breakout method for escaping from local minima. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*, pages 40–45, 1993.
38. J.P. Müller. The right agent (architecture) to do the right thing. In J.P. Müller, M.P. Singh, and A.S. Rao, editors, *Intelligent Agents V*, Lecture Notes in Artificial in Artificial Intelligence, Vol. 1555, pages 211–226. Springer-Verlag, Berlin et al., 1999.
39. D. Musliner and B. Pell (Cochairs). Agents with adjustable autonomy. Papers from the AAAI spring symposium. Technical Report SS-99-06, AAAI Press, Menlo Park, CA, 1999.
40. A. Narayek. Constraint-based agents. In E.C. Freuder, editor, *Constraints & agents. Papers from the AAAI Workshop (Technical Report WS-97-05)*, pages 45–50. AAAI Press, Menlo Park, CA, 1997.
41. A. Narayek. A planning model for agents in dynamic and uncertain real-time environments. In *Proceedings of the AIPS Workshop on Integrating Planning,*

Scheduling and Execution in Dynamic and Uncertain Environments, pages 7–14, 1998.

42. A. Roth, J. Murnighan, and F. Schoumaker. The deadline effect in bargaining: Some experimental evidence. *American Economic Review*, 78:806–823, 1988.
43. S.J. Russell. Rationality and intelligence. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, 1995.
44. S.J. Russell and E.H. Wefald. *Do the right thing: Studies in limited rationality.* The MIT Press, Cambridge, Mass., 1991.
45. S.J. Russell and E.H. Wefald. Principles of rationality. *Artificial Intelligence*, 49(1-3):361–395, 1991.
46. T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohmé. Anytime coalition structure generation with worst case guarantees. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, pages 46–53, 1998.
47. T. Sandholm, S. Sikka, and S. Norden. Algorithms for optimizing leveled commitment contracts. In *Proceedings of 1999 International Joint Conference on Artificial Intelligence (IJCAI-99)*, 1999.
48. T. Sandholm and N. Vulkan. Bargaining with deadlines. In *Proceedings of 16th National Conference on Artificial Intelligence (AAAI-99)*, 1999.
49. T.W. Sandholm and V.R. Lesser. Issues in automated negotiation and electronic commerce: Extending the contract net framework. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 328–335, 1995.
50. T.W. Sandholm and V.R. Lesser. Advantages of a leveled commitment contracting protocol. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, pages 126–133, 1996.
51. A. Sathi and M.S. Fox. Constraint-directed negotiation of resource reallocations. In M.N. Huhns and L. Gasser, editors, *Distributed Artificial Intelligence, Volume 2*, pages 163–193. Pitman/Morgan Kaufmann, Cambridge, MA, 1989.
52. G. Solotorevsky, E. Gudes, and A. Meisels. Modeling and solving distributed constraint satisfaction problems. In *Proceedings of the 2nd International Conference on Principles and Practice of Constraint Programming (CP-96)*, pages 561–562, 1996.
53. K. Sycara, S. Roth, N. Sadeh, and M. Fox. Distributed constrained heuristic search. *IEEE Transactions on Systems, Man, and Cybernetics*, 26(1):1446–1461, 1991.
54. G. Verfaillie and T. Schiex. Solution reuse in dynamic constraint satisfaction problems. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, pages 307–312, 1994.
55. T. Wagner, A. Garvey, and V.R. Lesser. Satisficing evaluation functions: The heart of the new design-to-criteria paradigm. Technical Report 1996-82, Computer Science Department, University of Massachussetts at Amherst, 1996.
56. T. Wagner, A. Garvey, and V.R. Lesser. Complex goal criteria and its application in design-to-criteria scheduling. In *Proceedings of the 1997 National Conference on Artificial Intelligence (AAAI-97)*, pages 294–301, 1997.
57. T. Wagner, A. Garvey, and V.R. Lesser. Criteria-directed task scheduling. Technical Report 1997-59, Computer Science Department, University of Massachussetts at Amherst, 1997.
58. T. Wagner and V.R. Lesser. Relating quantified motivations for organizationally situated agents. In *Proceedings of the 6th International Workshop on Agent Theories, Architectures, and Languages (ATAL-99)*, 1999.

59. T. Wagner, A. Raja, and V.R. Lesser. Modeling uncertainty and its implication on design-to-criteria scheduling. Technical Report 1999-51, Computer Science Department, University of Massachussetts at Amherst, 1999.
60. R.J. Wallace and E.C. Freuder. Anytime algorithms for constraint satisfaction and SAT problems. *SIGART Bulletin*, 7(2), 1996.
61. G. Weiß. On building flexible agents. Technical Report FKI-236-00, Institut für Informatik, Technische Universität München, 2000.
62. M.J. Wooldridge. Intelligent agents. In G. Weiss, editor, *Multiagent Systems*, pages 27–77. The MIT Press, Cambridge et al., 1999.
63. M.J. Wooldridge and N.R. Jennings. Agent theories, architectures, and languages: A survey. In M.J. Wooldridge and N.R. Jennings, editors, *Intelligent Agents*, Lecture Notes in Artificial in Artificial Intelligence, Vol. 890, pages 1–39. Springer-Verlag, Berlin et al., 1995.
64. M. Yokoo. Asynchronous weak-commitment search for solving distributed constraint satisfaction problems. In *Proceedings of the First International Conference on Principles and Practice of Constraint Programming (CP-95)*, pages 88–102, 1995.
65. M. Yokoo, E. Durfee, T. Ishida, and K. Kuwabara. Distributed constraint satisfaction for formalizing distributed problem solving. In *Proceedings of the Twelfth IEEE International Conference on Distributed Computing Systems*, pages 614–621, 1992.
66. M. Yokoo and K. Hirayama. Distributed breakout algorithm for solving distributed constraint satisfaction problems. In *Proceedings of the 2nd International Conference on Multi-Agent Systems (ICMAS-96)*, pages 401–408, 1996.