Task Management under Change and Uncertainty
Constraint Solving Experience with the CALO Project

Pauline M. Berry, Karen Myers, Tomás E. Uribe, and Neil Yorke-Smith

Artificial Intelligence Center, SRI International, Menlo Park, CA 94025, USA.
{berry,myers,uribe,nysmith}@ai.sri.com

Abstract The goal of CALO (Cognitive Assistant that Learns and Organizes) is to design an automated personal assistant to support a busy high-level knowledge worker. Operating in an inherently dynamic and uncertain domain, CALO relies on constraint technology in several components of its architecture. We outline the challenges and opportunities presented by constraint solving in the presence of change and uncertainty, embodied in CALO's personalized time management and task reasoning and execution systems.

1 Introduction

Consider the challenge of developing intelligent agents that assist a human by performing and managing tasks on her behalf. Such agents, operating in the real world, must handle change and uncertainty as a matter of course. This is the challenge of developing *CALO* (Cognitive Assistant that Learns and Organizes), an automated assistant hat will support a busy knowledge worker, such as a lab director or senior manager.[1] CALO should be able to perform routine office tasks for its user (e.g., arrange meetings, complete online forms, file email), manage open-ended processes (e.g., purchase a computer, arrange a conference), and anticipate and act on future needs of its user.

The cognitive assistant domain is inherently dynamic and uncertain. The user herself works in the evolving real world, with limited knowledge; with estimated, outdated or inaccurate information; interacting with colleagues; and encountering events outside her control, sometimes unpredicted. Moreover, users may change their minds, have inconsistent or mutually unachievable desires, and not be able or willing (or have the time) to communicate with their assistants.

In the following sections, we concentrate on two components of CALO where change and uncertainty impact the use of constraints technology: the personalized time management system, and the task reasoning and execution system.

Advantages of Constraints. We have found that using constraints in a system like CALO has a number of natural advantages. (1) Constraints have a well-defined, often intuitive semantics, which can be aligned with the ontology used in the system's knowledge base. (2) Constraints can capture qualitative and quantitative preferences and costs. (3) Constraints offer a declarative representation that is easy to produce and consume by different modules, including the human interface. (4) Constraints support notions of relaxation and explanation. (5) Finally, of course, constraints are supported by a large set of algorithms, solvers, and tools.

[1] www.ai.sri.com/project/CALO

2 Personalized Time Management

Time management is intensely personal. Many people, especially busy knowledge workers, are reluctant to relinquish control over the management of their own time. Moreover, individuals have different preferences and practices regarding how they schedule their time, how they negotiate appointments with others, and how much information they are willing to share when doing so. They also have different needs and priorities regarding the reminders they should receive.

The Personalized Time Manager (*PTIME*) assistant [1] is a CALO component with the goal of managing an individual's temporal commitments over an extended period of time, while recognising and adapting to the differences between individuals. Interaction between the human user and the system is central to this goal. The scheduling solutions found by the system should be informative, and the user–system dialogue should improve the quality of future interactions.

Requirements. At the heart of PTIME is a constraint-based scheduler. Solving the basic constraint problem is straightforward for small and medium instances, given present scheduling practice and available constraint technology. However, the basic problem is complicated by a number of factors, many due to change or uncertainty in one form or another. The following are some of the requirements for PTIME that go beyond pure constraint solving.

User requests At any moment the user can present new calendaring requests to PTIME. These include scheduling a new meeting, rescheduling existing ones, or adjusting locations and participants, while minimising perturbation on the user's schedule.

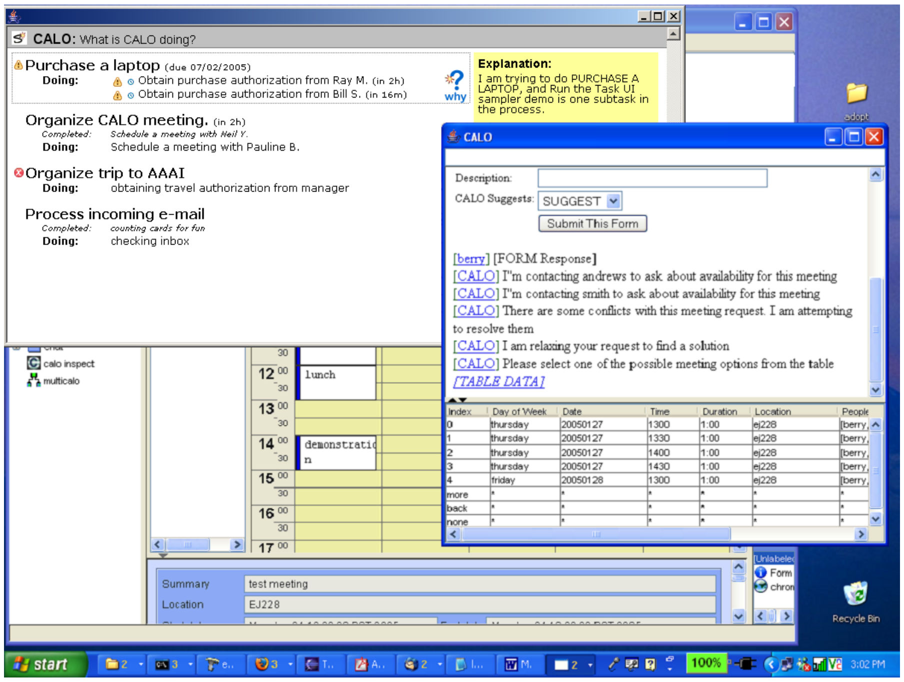
Preferences As described above, individuals have intense preferences over their time. These encompass not only when to schedule meetings—preferences on temporal and non-temporal constraints—but also the amount of autonomy the user permits PTIME. The user may not be able to articulate all her preferences, and may not be able or willing to communicate them to CALO; thus information is incomplete.

Execution Not everything occurs as planned: travel is delayed, meetings overrun, resources become unavailable. From the information CALO acquires about current execution, PTIME must reschedule, with the criteria of minimising perturbation and maximising likelihood of successful execution.

Relaxation Frequently, a meeting request cannot be satisfied without relaxing some constraints or violating some of the user's preferences.

Explanation An important requirement for CALO is that it be able to explain its own behaviour to its user. When a request cannot be satisfied, PTIME should be able to explain why this was the case, and present alternatives.

Interactivity The PTIME interface, shown in Figure 1, presents a view of the user's calendar, together with an interface for mixed-initiative decision making. For instance, if a meeting request is unsatisfiable, deciding whether and how to relax the request becomes a dialogue between the system and the user [2].



CALO: What is CALO doing?

Purchase a laptop (due 07/02/2005)
Doing: Obtain purchase authorization from Ray M. (in 2h)
Obtain purchase authorization from Bill S. (in 16m)

Explanation:
I am trying to do PURCHASE A LAPTOP, and Run the Task UI sampler demo is one subtask in the process.

Organize CALO meeting. (in 2h)
Completed: *Schedule a meeting with Neil Y.*
Doing: Schedule a meeting with Pauline B.

Organize trip to AAAI
Doing: obtaining travel authorization from manager

Process incoming e-mail
Completed: *counting cards for fun*
Doing: checking inbox

CALO

Description:
CALO Suggests: SUGGEST

[berry] [FORM Response]
[CALO] I"m contacting andrews to ask about availability for this meeting
[CALO] I"m contacting smith to ask about availability for this meeting
[CALO] There are some conflicts with this meeting request. I am attempting to resolve them
[CALO] I am relaxing your request to find a solution
[CALO] Please select one of the possible meeting options from the table
[TABLE DATA]

Index	Day of Week	Date	Time	Duration	Location	People
0	thursday	20050127	1300	1:00	ej228	[berry,
1	thursday	20050127	1330	1:00	ej228	[berry,
2	thursday	20050127	1400	1:00	ej228	[berry,
3	thursday	20050127	1430	1:00	ej228	[berry,
4	friday	20050128	1300	1:00	ej228	[berry,
more	*	*	*	*	*	*
back	*	*	*	*	*	*
none	*	*	*	*	*	*

Summary	test meeting
Location	EJ228

Figure 1. Development view of the PTIME user interface

Learning Over time, PTIME must learn and adapt to the user's preferences, which may themselves change. At present, PTIME can learn from explicit instruction ("I prefer morning meetings, except on Monday"), or implicitly, from the user's selection from a set of presented solutions to the CSP ("I prefer schedule A to schedule B") [3].

Distributed scheduling By its nature, satisfying a multi-person meeting request is a distributed negotiation problem over each potential participant. PTIME faces incomplete information on the preferences and schedules of others.

Solutions. The present constraint model in PTIME consists of prioritised finite domain constraints (FD), and disjunctive temporal constraints with semiring-based preferences (DTPP) [9]. There is a multi-criteria objective function, giving a soft constraint optimisation problem (COP). A sequence of COPs is solved, as the user interacts with the system to meet a calendaring request to her best satisfaction.

Although the COP is not large in size, and the mixed-initiative setting permits up to 500ms, say, for its solving, efficiency is important. This is due to the multiple COPs that must be solved for relaxation, explanation, negotiation, and rescheduling. Currently, solving is achieved by an FD-DTPP hybrid within a branch-and-bound search.

The present approach to change and uncertainty in PTIME is largely reactive. Our future work is to provide more robust schedules by proactive reasoning over the execution of tasks. In particular, we are considering learned probabilities on meeting occurrences and expected durations, and models of contingent events [6,11].

3 Task Reasoning and Execution

Ultimately, CALO assists its user by performing tasks on her behalf. These can range from keeping track of the user's commitments and providing reminders, to collaborating with the user in a mixed-initiative fashion, to fully autonomous completion of tasks explicitly delegated by the user, to proactive undertaking of activities that CALO determines will be beneficial (according to the permitted level of adjustable autonomy).

At the heart of CALO's ability to act is a *Task Manager* that initiates, tracks, and executes activities and commitments on behalf of its user, while remaining responsive to external events. The Task Manager component of CALO is based on a reactive execution system called *SPARK* [5].

SPARK is an agent framework grounded in a model of procedural reasoning. It is based on the Belief-Desire-Intention (BDI) model of agency [10], which has become the predominant architecture for the design of cognitive agents. SPARK has been developed to support the construction of large-scale, practical agent systems, and contains sophisticated mechanisms for encoding and controlling agent behaviour. At the same time, SPARK has a well-defined semantic model that is intended to support reasoning about the agents' knowledge and execution in a dynamic environment.

Constraint technology enables the temporal and resource reasoning of the Task Manager, together with parts of SPARK's deliberation over cognitive states. Both operate under the same requirements listed above for PTIME. Ongoing work includes:

Reasoning over commitments Action depends on knowing how to act, having the necessary means, and having sufficient time. Tasks are represented as hierarchical process models with simple deadlines. Richer temporal information will be represented as a Simple Temporal Network, augmented with information on expected (remaining) duration, probability of success, contingent events [6], and a profile of resource usage [4]. An important challenge is rapid incremental computation, suited to SPARK's reactive execution paradigm and the soft real-time demands on CALO's responses.

Guidance and goal selection Action follows decision on what to do. The Task Manager receives potential goals from the user, or raises them proactively. These, together with their deadlines and resource requirements, must be balanced against existing commitments in the light of user-stated *guidance* [8]. The various criteria may mutually conflict. The deliberation is modelled as a soft multi-criteria constraint optimisation problem over SPARK's cognitive states and the set of potential and current goals [7].

4 Conclusion

Change and uncertainty are real and pressing aspects for a cognitive assistant such as CALO. They arise not only from the situations in which CALO operates, but also from the demands of its user. Constraint-based models and constraint solving provide key functionality for several components of the CALO architecture. In such a context, we must address dynamism, evolving requirements, incomplete knowledge, and ill-known outcomes head on. The case is made for sustained research in these areas, to develop more expressive CSP-based modelling frameworks and solving algorithms; and for systems that make these advances available in practice.

The cognitive assistant domain features other aspects beyond change and uncertainty that are just as demanding of the constraint solving. These include execution in the real world, mixed-initiative problem solving and interactivity, user preferences, learning, and distributed reasoning. Here again constraint technology is evolving. The case is made for an effort to provide constraint systems—whether languages or toolkits—that can help simultaneously address all these aspects.

Acknowledgement. We recognise all the members of the CALO project, and thank K. Conley for technical assistance. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under Contract No. NBCHD030010.

References

1. P. Berry, M. Gervasio, T. E. Uribe, M. E. Pollack, and M. D. Moffitt. A personalized time management assistant. In *AAAI 2005 Spring Symposium Series*, Stanford University, CA, Mar. 2005.
2. P. Berry, M. Gervasio, T. E. Uribe, and N. Yorke-Smith. Mixed-initiative issues for a personalized time management assistant. In *Proc. of ICAPS'05 Workshop on Mixed-Initiative Planning and Scheduling*, pages 12–17, Monterey, CA, June 2005.
3. M. T. Gervasio, M. D. Moffitt, M. E. Pollack, J. M. Taylor, and T. E. Uribe. Active preference learning for personalized calendar scheduling assistance. In *Proc. of the 2005 Intl. Conf. on Intelligent User Interfaces (IUI'05)*, San Diego, CA, Jan. 2005.
4. P. Laborie. Algorithms for propagating resource constraints in AI planning and scheduling: existing approaches and new results. *Artificial Intelligence*, 143(2):151–188, 2003.
5. D. Morley and K. Myers. The SPARK agent framework. In *Proc. of the Third Intl. Joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS'04)*, pages 714–721, New York, NY, July 2004.
6. P. Morris, N. Muscettola, and T. Vidal. Dynamic control of plans with temporal uncertainty. In *Proc. of IJCAI'01*, pages 494–502, Seattle, WA, Aug. 2001.
7. K. Myers and N. Yorke-Smith. A cognitive framework for delegation to an assistive user agent. In *AAAI 2005 Fall Symposium Series (to appear)*, Arlington, VA, Nov. 2005.
8. K. L. Myers and D. N. Morley. Policy-based agent directability. In H. Hexmoor, R. Falcone, and C. Castelfranchi, editors, *Agent Autonomy*, volume 7 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, chapter 9. Springer, 2003.
9. B. Peintner and M. E. Pollack. Low-cost addition of preferences to DTPs and TCSPs. In *Proc. of AAAI-4*, pages 723–728, San Jose, CA, 2004.
10. A. S. Rao and M. P. Georgeff. Modeling agents within a BDI-architecture. In *Proc. of KR'91*, pages 473–484, 1991.
11. K. B. Venable and N. Yorke-Smith. Disjunctive temporal planning with uncertainty. In *Proc. of IJCAI'05*, Edinburgh, UK, Aug. 2005.