Dynamic Workflow Verification for Health Care

Jeff Dallien[†⋆], Wendy MacCaull[†⋆⋆], Allen Tien[‡]

[†]Department of Mathematics, Statistics and Computer Science, St. Francis Xavier University, Antigonish, NS. http://logic.stfx.ca/
[‡]Medical Decision Logic, Inc., Towson, MD. http://www.mdlogix.com/

1 Introduction

This research is initial work in the design and development of verifiable Workflow Management Systems (WfMS) for application to biomedical research, clinical care and public health. Though many software products are sold as workflow systems, existing products are weak in 3 areas: (i) representation of workflow rules; (ii) dynamic adaptation; and (iii) verification, all of which are critical in the health sector. Solving the conceptual and technological issues in the workflow problem has significant economic value, not the least of which is efficiency and effectiveness of patient care. Indeed, this is part of a research project that addresses five of the ten key recommendations in the Canadian Institutes of Health Research national consultation [4], namely: (i) workforce planning; (ii) management of the healthcare workplace; (iii) timely access to care; (iv) managing for safety and quality; and (v) managing and adapting to change. Here we describe some applications and progress in developing an automated verifiable dynamic WfMS. Interfacing such tools with Canada's emerging electronic health records will lead to significant commercialization opportunities.

2 Applications of WfMS in Health Care

Workflow is defined as the movement of tasks through a work process describing how tasks are structured, who performs them, and their relative order. The following details two applications of WfMS for health.

In a drug trial, each task to be performed is added to a workflow to describe to medical staff what must happen at each point in the trial; eg., administering drugs, taking readings, scheduling appointments, etc. Currently the planning is done manually, perhaps with the assistance of a computer package to help draw and store the workflow. Situations could arise during a drug trial that should be addressed before patients receive treatment. For example, a patient may be assigned incompatible treatments, a cycle in the workflow may arise by assigning a patient from one part of the drug trial to another and back again, or a patient

⋆ Support from an NSERC Research Capacity Development Graduate Fellowship is gratefully acknowledged.

⋆⋆ Support from Science and Engineering Research Council of Canada is gratefully acknowledged.

may reach a point in the drug trial where no rules apply to them. Such situations cost time and money and could adversely affect the patient.

Hospice palliative care is a combination of therapies that address the physical, psychological, social, spiritual and practical needs of individuals with chronic or terminal illnesses. Care is provided in numerous settings involving a complex network of health professionals, and others. Tools are needed to guide and inform the trajectory of patient care, be inclusive of all participants and care settings and adapt dynamically to changes without compromising the quality and safety of care. With verifiable tools, the complex network can be analyzed to determine if desirable outcomes are always met and undesirable ones never occur.

3 Related Work

Many WfMSs have been developed but emphasis has not been placed on verification of the workflow. AgentWork [8] uses dynamic rules to allow users to identify possible problems; the WfMS will adapt the workflow at runtime based on the actions described in the dynamic rules. An example from [8] is used to remove a drug from a patient's treatment based on the results of a blood test until the reading returns to normal:

WHEN critical-blood-status VALID-TIME (now - (5, day), now) AND
present-in-further-workflow(Drug-Administration(drug, "Etoposid", P)
THEN drop(Drug-Administration(drug, "Etoposid", P)
Unless normal-blood-status(P) VALID-TIME now

Such rules are valuable because they allow the WfMS to handle situations that are exceptional, but more is needed. Analysis of the system and all the possible scenarios is required as this is a safety critical system. While rules can be used to add a safety net to the workflow, many possible workflows result due to the combinations of rules which could be triggered. It is possible, even likely in complicated systems, that the planners do not understand all of the possible scenarios which could be created by these rules being triggered.

The METEOR system [1] is designed to manage workflows over large distributed systems. It also supports dynamic changes and exception handling.

Neither of these systems support verification as an analysis tool in the planning stage of a workflow. They rely on catching the errors at runtime and then dealing with them. This can make designing a workflow easier, but verification can help discover problems before the workflow is put into action. One existing "workflow diagnosis tool" is Woflan [12]. It is used for analysis during planning, but it cannot verify arbitrary properties of a workflow. It is limited to checking "soundness" of a workflow, i.e., there is no deadlock and no unreachable tasks.

4 Dynamic Adaptation and Verification

Our verification system is designed as an analysis tool to discover where in a dynamic workflow problems could occur before the drug trial or patient care begins. Dynamic rules are good at handling safety properties, that is, something bad does not happen. A rule can prevent a potential problem by changing the workflow based on new data but cannot ensure that something desirable will eventually happen or always happens because the time at which the rules are triggered is uncertain and they are not intended to analyze the workflow as a whole. It is possible that the change to the workflow caused by a rule could create a new, undesirable situation. In a rule-only based system, another rule would need to be triggered to fix the problem, which could then cause another problem, requiring another rule and so on.

The goal of our system is to allow workflow designers to check both liveness and safety properties of a workflow, including the possible situations caused by dynamic rules during runtime. It avoids the rules-fixing-rules problem and does not require the workflow system to expend the runtime resources required to check rules needed only to catch problems caused by other rules. A verification tool like ours can be run offline once to check a property and only needs to be run again when new dynamic rules are added or the base workflow is changed. A tool that can check workflows and the effect of dynamic rules will allow designers to produce a workflow and rules package that is known to be correct.

5 System Details

Input to the system will be the workflow plan, the set of dynamic rules, and the properties to be verified. The system will then check that these properties hold on the original workflow and also any workflow that could be created by applying the rules for dynamic changes.

Description Logics (DL), knowledge representation languages that allow for expression and reasoning about structured knowledge [2], form the framework within which our syntax was developed. A typical DL knowledge base consists of a set of terminiological definitions, known as the TBox, which uses unary and binary predicates (called concepts and roles) and expresses how they are related. The knowledge base also includes an assertional part, the ABox, which identifies individuals in the system based on the definitions in the TBox.

The system we are working on will support the following:

1. A knowledge base using type 0 ontologies [11] in Coherent Description Framework to express what is happening at each state of the workflow. The medical community has made extensive use of ontologies for representing terminologies and the relationships that exist between the objects in the terminologies. Large medical ontologies have been developed by various organizations such as Health Level 7 [6] to create standards for the exchange, management and integration of health care information. Ontologies may be incorporated into the knowledge base of the theorem prover allowing us the use of both explicit and implicit knowledge included in these existing medical ontologies. For example, the verification tool may be used to check that drugs from two classes which are known to be incompatible are not administered simultaneously without explicit listing of all of the pairs of drugs from the two classes.

2. Support for dynamic changes to the workflow with rules similar to those used in [8]. The theorem prover will check the effect of the rules being triggered on the properties that are being checked. For complexity reasons, the dynamic rules are only triggered after key events that would add data to trigger a rule. Currently, simple dynamic rules are implemented in the theorem prover, such as adding a single occurance of a new task, or adding task B whenever task A occurs. Work is in progress to deal with complex dynamic rules such as adding a repetitive task when the portion of the workflow affected also contains a repetitive task. The problem is adding the new loop to the existing one when the loops may be of different lengths and/or time granularities.

3. Temporal Description Logic ($\mathcal{ALCT}$) [10] for expressing properties of the workflow to be checked by the theorem prover. This logic is a basic DL with the addition of LTL temporal operators which can be used on concepts. While this does not provide the level of detail of a system that allows references to specific days of the week, $\mathcal{ALCT}$ is decidable. This is important since our system will be checking all possible paths and active rule effects up front. We can verify properties like "Patients in trial arm A will be under the care of physician B until the trial ends" and "Follow up visits are scheduled before discharge." Other benefits of using $\mathcal{ALCT}$ are that DLs have a close relationship to ontologies which are already used in our system and in the health care field [7]; and DL incorporates a syntax designed for the non-specialist in logic, facilitating the use of the system. Research into identifying commonly used patterns of formulas may be found in [9] and [3].

The implementation of the verification tool is a tableau-style theorem prover [5] written in XSB Prolog using the Coherent Description Framework extension, which permits the integration of ontologies. The verification tool is being designed to be compatiable with a WfMS being created by Medical Decision Logic, Inc. This workflow engine supports dynamic rules like those used in the verification tool and uses ontologies through the CDF extension of the XSB language. The goal of developing our own reasoning engine using CDF and XSB is to be compatible with this system and to take advantage of the $\mathcal{ALCT}$ language, which we did not find implemented in any existing tools.

6 Future Work

We are working to include some measure of actual clock time in the workflow model, so the active rules can indicate a duration for new tasks being added. The smallest time unit considered so far is one hour. By incorporating actual time, we can check properties like "Severe pain will be addressed within 2 hours."

The complexity of the verification problem for large systems may make analysis intractable. We plan to investigate compositionality-based methods to find a way of dividing the verification problem into smaller pieces while still guaranteeing correctness in the larger system.

References

1. K. Anyanwu, A. Sheth, J. Cardoso, J. Miller, and K. Kochut. Healthcare enterprise process development and integration. Technical report, LSDIS Lab, Department of Computer Science, University of Georgia, 2002.
2. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *Description Logic Handbook*. Cambridge Univeristy Press, 2003.
3. J. Dallien and W. MacCaull. Automated recognition of stutter invariance of LTL formulas. *Atlantic Electronic Journal of Mathematics*, to appear, 2006.
4. M. Dault, J. Lomas, and M. Barer. Listening for Directions II. Final Report, 2004. Canadian Health Services Research Foundation.
5. M. Fitting. *First Order Logic and Automated Theorem Proving*. Springer, 1995.
6. Health Level 7. http://www.hl7.org/.
7. M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented systems and frame-based languages. *Journal of the ACM*, 42:781–843, 1995.
8. R. Müller, U. Greiner, and E. Rahm. Agentwork: a workflow system supporting rule-based workflow adaptation. *Data & Knowledge Engineering*, 2004.
9. D. Păun and M. Chechik. On closure under stuttering. *Formal Aspects of Computing*, 14(4):342–368, 2003.
10. K. Schild. Combining terminological logics with tense logic. In *Proceedings of the 6th Portuguese Conference on Artificial Intelligence*. EPIA '93, October 1993.
11. T. Swift. The Meaning of Cold Dead Fish. http://www.cs.sunysb.edu/~tswift/.
12. E. Verbeek and W.M.P. van der Aalst. Woflan 2.0: A petri-net-based workflow diagonsis tool. Lecture Notes in Computer Science, vol. 1825. Springer, 2000.