In Preparation for Joint Action

Susan L. Epstein
Hunter College / CUNY
Department of Computer Science
New York, NY
sepstein@hunter.cuny.edu

Matthew Evanusa
Hunter College / CUNY
Department of Computer Science
New York, NY
matthew.evanusa@gmail.com

Anoop Aroor
The Graduate Center / CUNY
Department of Computer Science
New York, NY
aaroor@gc.cuny.edu

ABSTRACT
This paper recommends extensive, human-like preparation for joint action. Given the substantive differences in the ways that people and robots perceive and reason, it argues that this knowledge be shared explicitly. Moreover, it assigns responsibilities to each agent based on their strengths and weaknesses. The approach detailed here deliberately explores to acquire knowledge about the specific task it confronts and the skills of the agents in that setting. It then exploits that knowledge to support performance.

Categories and Subject Descriptors
[**Human-centered computing**]: Collaborative and social computing theory, concepts and paradigms --- computer supported cooperative work.

General Terms
Design, Reliability, Experimentation, Human Factors

Keywords
joint action, discovery, scribe, task allocation

1. INTRODUCTION
In *joint action*, a team of agents works together on a common task. Each agent's skill at the actions it undertakes has considerable impact on the team's performance. The thesis of this paper is that joint action produces a better outcome when the agents uncover and share knowledge beyond the problem description. The approach described here poses questions about the objects involved, and has the agents find and share answers. This paper postulates roles for agents based on what they do best. Its principal contributions are an explicit, robot-supported representation for inter-agent communication, and a human-guided discovery period. Together, they support subtask allocation that maximizes task expertise, in preparation for joint action.

Table 1: A scenario in preparation for joint action

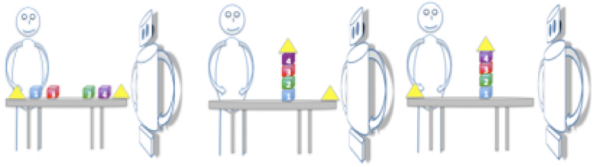
1. What do and don't we know?	Initialize board
2. What do we want to achieve?	Post goal state
3. How shall we do that?	Post intentions
4. How can we act?	Action repertoire
5. What can we detect?	Sense
6. How hard is this?	Discover
7. How shall we proceed?	Subtask allocation

Conference'10, Month 1–2, 2010, City, State, Country.

The scenario in Table 1 prepares intelligent agents for joint action [1]. It is intentionally reminiscent of how two people might address a challenging assembly task together, for example, building an unwieldy piece of IKEA furniture. They open the package, lay out the contents, and look at the picture of the final product. The directions identify the parts and a sequence of actions to execute. Then the people's respective skills (e.g., reading, spatial cognition, accuracy, agility, strength, dexterity) come into play as they work through the assembly together. Each step in Table 1 poses a question in preparation for the agents' pursuit of a common goal. These questions and their answers are the focus of this paper.

We address our thesis in the requested framework for this workshop, which we call task *T*. A person and a robot are asked to construct a pile of four cubes in a pre-specified order, and to top the pile with a pyramid. Figure 1 shows the start state and the two possible goal states. If it is within reach, an agent can *take* an *object* (i.e., a cube or a pyramid) from the table or from the pile, *put* an object on the pile, *give* an object to the other agent, or *support* the pile so that it is less likely to fall. Both agents know the location of each object and which of them can reach which objects. At any point in time, an object is on the table, in the pile, or *held by* (in the grip of) one of the agents.

The crucial differences between the two agents are highlighted here by consideration of a simplified task. *T′* is a thought experiment where all blocks are reachable and the robot is an (unembodied) computer. For the person alone, *T′* is simple, because people bring to bear commonsense and real-world experience. Naïve physics dictates that the pile must be constructed from the bottom up. The person would stack the bottom cube first, then the next, and so on. *Stack* is a macro-action, the sequence of actions <*take, put*>. The person may never consciously plan at all, but merely produce reactive *stack* intentions as she responds to differences between the start state and goal state. To execute a *stack*, the person expands it to <*take, put*>. For the computer alone, however,



(a) (b) (c)

Figure 1: (a) the start state; (b) and (c) the goal states.

even *T′* is complex. The computer lacks the person's ability to focus attention on cube 1 first, so it might flounder about in a large search space. Of course, if the computer had preconditions and add-delete lists for each action, it could plan. An ordered, ABSTRIPS-like sequence might emerge: pile cube 1, cube 2, cube 3, cube 4, and finally a pyramid [2]. Then the computer could expand each of those steps, either in advance or as it constructs the pile. That requires the ability to transform an action into a sequence of micro-actions (e.g., *take* = <*reach, grasp, lift*>).

The original task, *T,* requires joint action because neither agent can perform it alone; neither of them can reach all the objects required for the pile. The person may not initially even consider this difficulty, but react to it only when it arises. If the computer plans, however, it will recognize that the person must *put* cubes 1 and 3 or *give* them. (A partial plan would accommodate this.)

Task *T* also requires real-world action. People learn *operationalizations* of the actions for *T* (the physical movements expected to achieve them) in infancy, and practice them repeatedly on a broad range of objects. Usually, a person will be unaware of how she performs these actions and will assume that they succeed. Without human guidance, however, a robot's world experience can be uncertain and error-ridden. Operationalization chooses a complex sequence of motor commands that must be calculated and then transmitted explicitly. For example, consider *reach* on an accessible object. Control must choose which arm to use, how to maneuver that arm into place, and how to confirm by sensors that it has arrived there. Without confirmation, accumulated errors from a sequence of actions can lead to failure. While the person is optimistic, the robot is, of necessity, wary.

Thus task *T* involves two extremely different agents. The person is reactive, with good motor skills that have been compiled out, and a repertoire of macro-actions with multiple ways to expand them. Moreover, the person has the ability to contend with unanticipated failures and is accustomed to joint action, at least with other people. If the pile topples or a block is not within reach, all is not lost. The robot, in contrast, plans, calculates quickly and accurately, and confirms its actions carefully. The more actions, micro-actions, and operationalizations available for them, the longer the robot will take to decide.

To facilitate this discussion, we elaborate slightly on task *T*, as follows. Each agent has two hands. Each object has a location in space whose value can be determined by the action *find*. The value returned by *find* is in an allocentric coordinate system for the table, augmented by special values for situations where it is in an agent's hand, in the pile *P*, or *missing* (i.e., not found). The predicate *reachable*(*l*) specifies whether an agent can reach location *l*.

As described, task *T* leaves open many key questions. Some address what the agents can do physically, including their reach, strength, and agility. Others address the agent's preferences and crucial but unspecified properties of the objects. The answers to these questions are productive only if they are *public*, that is, known to both agents. A mechanism that shares those answers is therefore central to our approach. It is described in the next section to answer the first four questions in Table 1. Section 3 considers what can be sensed or discovered in advance, and Section 4 discusses how to assign subtasks to agents.

2. PUBLIC KNOWLEDGE

To share the knowledge highlighted by Table 1, we envision a representation that is a public knowledge board (henceforward, simply the *board*). An example appears in Figure 2, where three cubes have already been stacked and the robot is holding cube 4 with its first hand. The board contains both Table 1's questions and some of their answers. It records the agents' shared world state, their action repertoire, their goal, their approach, and their percepts. A *post* is an item on the board; each is either a *fact* (with a fixed value) or a *fluent* (with a changeable one). Initially, all facts and fluents are posted, the latter without values (step 1 of the scenario). Here, facts are the label and shape of each object, and the number of hands each agent has; fluents are each object's location, what is in each hand, and what is within reach of each agent.

The board includes the goal (step 2 of the scenario). Our approach addresses one subgoal (in boldface) at a time. Subgoals already accomplished are checked off, and the current subgoal appears in boldface. There is also a *sketch*, an ordered list of macro-actions to achieve the goal (step 3 of the scenario). The sketch is a compromise between the robot's reliance on detailed plans and the person's reactivity. Each macro-action can be thought of as an

Facts	Fluents		Goal	Sketch	Status
cube(Obj1)	*location*(Obj1) = *P* (R)	*holding*(HH1) = pile (H)	*location*(Obj1) = *P* ✓	*stack*(Obj1) ✓	Agent R
cube(Obj2)	*location*(Obj2) = *P* (R)	*holding* (HH2) = pile (H)	*location*(Obj2) = *P* ✓	*stack*(Obj2) ✓	*take*(Obj4) ✓
cube(Obj3)	*location*(Obj3) = *P* (R)	*holding* (RH1) = Obj4 (R)	*location*(Obj3) = *P* ✓	*stack*(Obj3) ✓	***put*(Obj4, *P*) (RH1)**
cube(Obj4)	*location*(Obj4) = (2,6) (R)	*holding* (RH2) = nil (R)	***location*(Obj4) = *P***	***stack*(Obj4)**	
pyramid(Obj5)	*location*(Obj5) = (1,9) (R)	…	*location*(*x*) = *P*	*stack*(*x*)	Agent H
pyramid(Obj6)	*location*(Obj6) = (3,7) (R)	*reachable*(*P*) = T (H)	& *pyramid*(*x*)	& *pyramid*(*x*)	*support*
hand(human,HH1)	…	*reachable*(*P*) = T (R)			
hand(human,HH2)		…			
hand(robot,RH1)					
hand(robot,RH2)					

Actions	Preconditions	Reliability % H	Reliability % R	Speed H	Speed R	Dexterity H	Dexterity R
take(*o*,*hand*)	*reachable*(*location*(*o*)), *graspable*(*o*), *liftable*(o), *holding*(*hand*) = nil	98.90	98.03	VG (H2)	G (H1)	G (H2)	VG (H1)
put(*o*,*hand*)	*holding*(*hand*) = *o*, *reachable*(*P*)	65.03	99.53	…	…	…	…
give(*o*,*hand*)	*holding*(*hand*) = *o, reachable*(*k*),	…	…				
support	*holding* (*hand1*) = *pile*, *holding*(*hand2*) = *pile*						
find(*o*)	*holding* (*hand1*) = nil, *holding*(*hand2*) = nil						

Figure 2: The board midway through task *T*. The robot *R* is about to *put* cube 4 on the pile *P* while the human *H* supports it.

unelaborated chunk of a plan. This deliberate lack of detail is motivated by the likelihood of error during task execution. For simplicity, the sketch here is STRIPS-like [3], but it could be from a planner or the person.) Accomplished macro-actions are also checked. The macro-action that addresses the current subgoal appears in boldface, and is expanded into an action sequence, with the current action in boldface. This answer to step 3 of the scenario dynamically focuses on a macro-action and an action sequence.

The board also includes the agents' action repertoire (step 4 of the scenario) with preconditions. (Note that there is no agent argument. An action is considered from the perspective of the agent that undertakes it.) Preconditions are based either on the person's commonsense input or the robot's first pass at operationalization. Figure 2, for example, indicates that if agent *a* is to *put* object *o* on the pile at *P*, it must be holding *o* and be able to rea*ch P*. Initially, preconditions are posted by the person, but the robot can save them for use in future tasks.

In the joint action envisioned here, before an agent undertakes an action, it announces its intent. Because both agents can err, it is necessary to confirm that every action produces the desired result (e.g., that the robot's hand is holding the block after a *take*). Confirmation appears as checks on the board in Figure 2. This would be a stylized language, for example, "Stacking cube 3. Success."

The board is the agents' shared worldview. It makes their intentions explicit and identifies actions underway in a display understood by both agents. Thus, it requires continual updates. The agent best suited for this task of *scribe* is the robot. Whatever the robot's internal representation for that knowledge, it can quickly direct error-free output there. Moreover, unlike a person, a robot is untroubled by repetition and can repeatedly scan its environment accurately and tirelessly. The scribe senses, confirms, and posts the results of all actions and the achievement of subgoals. It also posts any new or changing values (e.g., preconditions, fluents), as well as the information discovered in the next section.

3. DISCOVERY

Two kinds of discovery are envisioned for step 6 of the scenario: computational inference and empirical investigation. For the first, an agent may sense or calculate the value of a fluent. For example, although locations are originally unknown and fluent, any agent may detect or confirm locations with *find.* As another example, given the *reachable* predicate and the currently posted location *l* of object *o*, each agent can calculate whether or not *reachable*(*l*) is true for that agent. If so, that value is posted with the agent who provided it. In Figure 2, the robot has located all the objects, and each agent has reported on its own hands. For empirical investigation, however, it is necessary to consider agents' expertise.

The *expertise* of an agent with respect to an action, is its ability to accomplish that action quickly and accurately [4]. Ideally, each agent would be expert at each action it undertakes. Nonetheless, an agent may execute an action poorly for all instantiations (e.g., the person's hands may be too unsteady to pile objects) or the agent may have difficulty only with a particular class of instantiations (e.g., the robot finds pyramids may be awkward for the robot to grasp). In the worst case, an action may damage the task objects (e.g., the robot applies a grip so strong that it crushes pyramids).

The second kind of discovery is an empirical investigation of the agents' expertise. Rather than make assumptions about an agent's ability or an object's properties, we propose that each agent discover its own expertise levels for each action. First, each agent performs a sequence of brief tests of its ability to execute each action. The agent instantiates an action on different blocks and attempts it alone, with each hand and with both hands together. (Ideally, this exploration would be with objects identical to those in the task, all within reach of both agents.) A key element of discovery here is generalization over success or failure. If an action fails, for example, is it the fault of the designated hand? or is the object simply too heavy to lift with one hand? or with both hands? To obtain a reliable estimate, an agent that has difficulty with a particular instantiation should attempt it more times than one where she consistently succeeds. Once these tests are complete, agents attempt together actions that they failed alone. For example, if the person could not pile cubes four high by herself, she would try to do so while the robot supported the pile.

Purely random exploration, of course, may produce little of value. During discovery, therefore, the person, with her store of commonsense world knowledge, should assume the role of *guide.* The guide describes what she attempts, and the robot imitates (e.g., "I can stack blocks five high. Can you?") The robot as scribe can also document action sequences that the person frequently uses during discovery. For example, if the robot mentions that the person often follows a *take* with a *put* on the pile, the person could say, "Yes. Let's call that *stack*" and so a macro-action is detected and learned. The sketch is therefore not posted until discovery is completed.

Discovery produces expertise ratings. Possible facets of expertise include *speed* (elapsed time per attempt), *dexterity* (amount of movement), and *reliability* (likelihood of success). Some of those values, on common scales, appear in Figure 2. In addition, because further information on the agents' expertise may emerge once discovery is over and the task is underway, the robot as scribe should modify expertise values to reflect its recent performance. Although discovery is of necessity empirical and therefore subject to error, it provides useful information for subtask allocation, as described next.

4. SUBTASK ALLOCATION

As envisioned here, joint action assigns each subtask of the sketch to an agent. This *subtask allocation* is a constrained optimization problem along two dimensions: whether the agent is likely to execute the subtask properly, and the expertise with which it will do so. Consider, for example, the macro-action *stack*(Obj3). Initially, the person is the obvious agent for this because she can reach cube 3 and the robot cannot. If, however, her hands are unsteady and (as the expertise ratings indicate) she knocks over the pile quite often when she tries to *put* a block on it, the robot may be a better choice. If the robot is to stack cube 3, however, the person must *give* it to the robot first, a modification to the sketch.

Without an overseer, subtask allocation requires a mutually agreed upon policy. One possible allocation policy is an agreement that each agent is the actor whenever it can reach the locations involved. That would suffice in *T* until conflict arose over the pyramid. It is possible, however, to learn an allocation policy. FORR, a general architecture for learning and problem solving, solicits opinions from heuristics called *Advisors,* and maximizes a weighted combination of those opinions to select an action [5]. These weights reflect the Advisors' accuracy and are learned by reinforcement, based on experience [6]. A subtask allocation policy could be learned similarly. One Advisor would support the agent with the greatest probability of success on a subtask, another the most dexterous, and another the fastest. There would also be an Advisor that considered whether the agent could satisfy the preconditions immediately. Once weights were learned, voting would allocate subtasks based on experience.

5. DISCUSSION

In task *T,* joint action is a necessity because neither agent can reach all the cubes alone. Another motivation for joint action is that the agents' ability to work in parallel or to bring greater expertise to different subtasks might improve performance on some metric (e.g., speed or resource consumption). Both cases benefit from knowledge about the abilities of both agents in the context of the current task, as supported by discovery.

The notion of a scribe is a preliminary step toward parallel execution during joint action. Once subtasks are assigned, another way to parallelize is to interleave agents' respective action lists with a scheduling algorithm. Parallelization could also permit an agent to do two things at once, for example, steady the tower for the other agent with one hand while it picks up the next object it is assigned to stack. This approach, however, is fraught with internal conflict for a multi-armed agent, as well as for motion planning with respect to the other agent. This is why, in Figure 2, *find*'s preconditions forbid the agent to do so unless its hands are empty.

The description of discovery provided here is only a first step. People often envision more imaginative macros. Perhaps, for example, an agent could create a *sub-pile* of multiple objects (e.g., cube 4 atop cube 3) and then move the sub-pile onto the pile. Whether it is possible to move a sub-pile successfully, and whether it is worth the risk, are the kind of questions young children address in their play. How to encourage such discovery and the length of time to devote to it are open questions.

Support is necessary only if there is a risk that a *put* will fail. A conservative approach would be to have one agent support the pile whenever the other executes a *put.* As in Figure 2, this can be implemented as a precondition on *put* that the other agent is holding the pile with both hands. A more thoughtful approach would be to have an agent support the pile whenever discovery indicated that the agent about to *put* is likely to fail, within some agreed upon threshold. Discovery could also address whether an agent could adequately support the pile with one hand, and where support should be placed (e.g., on opposite sides of the pile? near the vertical center?) for the best result.

Support of the pile is an example of cautious behavior. The degree of caution during joint action should be specified in advance as a tolerance for unsuccessful actions. Discovery may address this too, so that a pyramid-crushing robot will be forbidden to *take* any pyramid, and an unsteady person will never stack at all. The board should also record other relevant properties of the objects (e.g., temperature, slipperiness).

Figure 2 is a metaphor, and should not be taken as a recommendation for logic as a representation. The board is intended to be accessible to both agents, but how to interpret its contents (e.g., *reachable, location*) should be agent-specific. For example, the agents in *T* represent space differently — the person's view is egocentric and qualitative, while the robot's is a real-valued, allocentric map [7]. The board may have to provide both. Moreover, how often to update the board, at what level of detail, and how to resolve differences of opinion about posts there are open questions. One way to address disagreement on the value of a fluent would be to consider how both agents might confirm, as people often do, the values of facts.

A post on the board, with its source, is an expression of an agent's belief. If the agents use the same computation (e.g., for speed), there should be no disagreement. If a fluent is sensed, however, there is the possibility that different values result. This leads to questions of authority. Who for example, knows best about what an agent is holding? The agent who feels the object or the one who observes it? One approach is to assign agents authority over specific fluents. Another is to have them both sense again, possibly in a different way, and compare the results.

If an action fails (e.g., the pile falls), the robot will correct the board accordingly. This includes unchecking accomplished subgoals and sketch entries so that joint action can resume. Some related issues remain, however. The assumption that one agent performs an action (e.g., they do not lift together) is restrictive. Moreover, the assumption that an agent, once assigned a task, accepts it and tries until it succeeds, may be unrealistic. The preference of the agents for a subtask may also be relevant to its allocation (e.g., the person may want to place the pyramid as a final, celebratory step, or the robot may need to recharge). In addition, several decisions in *T* should be the subject of negotiation, which is outside the scope of this paper. These include the location *P* of the pile, the subtask allocation policy, and the acceptable risk threshold for failure.

In summary, preparation for joint action assigns the roles of guide and scribe to the person and the robot, to capitalize on their respective strengths. Discovery gathers useful, empirical knowledge, and shares that knowledge through the board with both agents, to guide subtask allocation attuned to the agents' demonstrated expertise. Together the agents can then address their task more effectively, together.

6. ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under #IIS-1117000. We thank the students in Hunter's machine learning lab for their thoughtful suggestions.

7. REFERENCES

[1] [1] Epstein, S. L. Wanted: Collaborative Intelligence. *Artificial Intelligence*, 2212015), 36-45.

[2] [2] Sacerdoti, E. D. Planning in a Hierarchy of Abstraction Spaces. *Artificial Intelligence*, 5, 2 1974), 115-135.

[3] [3] Fikes, R. E. and Nilsson, N. J. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving, 21972), 189-208.

[4] [4] D'Andrade, R. G. *Some Propositions about the Relations between Culture and Human Cognition*. Cambridge University Press, City, 1990.

[5] [5] Epstein, S. L. Prior Knowledge Strengthens Learning to Control Search in Weak Theory Domains. *International Journal of Intelligent Systems*, 71992), 547-586.

[6] [6] Petrovic, S. and Epstein, S. L. Tailoring a Mixture of Search Heuristics. *Constraint Programming Letters*, 42008), 15-38.

[7] [7] Klatzky, R. L. *Allocentric and egocentric spatial representations: Definitions, distinctions, and interconnections.* . Springer, City, 1998.