Mind Readers and Mind Users: The Utility of Sharing Architectural Components across Multiple Robots

Matthias Scheutz

Tufts University, Medford, MA 02155, USA
matthias.scheutz@tufts.edu

Abstract. We introduce the concept of architectural "component-sharing" as the basis for "knowledge sharing" in hive minds, e.g., a system multiple robots connecting by shared components in their control architectures. We discuss the architectural requirements and demonstrate the utility for multi-robot instruction and automatic reasoning across robotic platforms with two examples of natural language human-robot interactions with mind-sharing robots. Finally, we discuss some of the challenges of making instructing hive minds intuitive for humans and point to questions that need to be addressed in the future.

Keywords: mind sharing, knowledge sharing, task-based dialogues, computational architecture

1 Introduction

Would it not be great at times if one could read another person's mind, not for nefarious purposes, but simple to coordinate joint activities? Consider a human team where team members are attempting to achieve a common goal. To coordinate their task-based activities they have to keep track of each other's mental states such as goals and subgoals, intentions, beliefs, and various other task-related aspects, i.e., they have to synchronize their individual "mental models" of their teammates to achieve overall common ground and common understanding of the moment-by-moment objectives. The synchronization, however, requires explicit effort and has to be accomplished through a mixture of observations, actions, inferences, and communicative interactions. If team members could telepathically transfer state updates directly to everyone's mind, the team would likely be more efficient in task execution and, moreover, not get into a state where the individual mental models are out of sync (which can lead to misunderstanding, wasted efforts, and overall reduced task performance).

While humans will not be able to enjoy such feats in the foreseeable future, such direct access to other minds is not a limitation for artificial agents like robots. So-called "hive minds" that share their all of their mental states, can be realized on robots and allow for a variety of capabilities that not only improve

the functions and operation of individual robots, but can also be a great utility in mixed human-robot teams (e.g., to support the formation and maintenance of "shared mental models" of human team members, i.e., the synchronization among individual mental models).

In this paper, we present the cognitive robotic DIARC architecture that allows for the implementation of hive minds by way of sharing architectural components across different architecture instances. Different from other cognitive architecture (e.g., SOAR, ACT-R, Epic, Prodigy, etc.) and robotic architectures (3T, RCS, etc.) where multiple instances can communicate with each other through special communication channels, the DIARC architecture can effortlessly share any number of components among multiple instances dynamically during task performance. We will describe the mechanisms by which such dynamic sharing can be accomplished and also demonstrate the utility of sharing various parts of the architecture for task learning, task execution, and support of shared mental models in mixed-initiative human-robot teams.

We start by motivating our approach to hive minds, followed by a brief overview of the DIARC architecture that includes the specific mechanisms that allow for component-sharing. We then focus on three multi-robot settings that demonstrate the utility of component-sharing for (1) multi-robot instructions and (2) automatic information gathering and sharing utilizing capabilities and information of other robots We finish by discussing opportunities and challenges of designing and using hive minds.

2 Motivation

Even though hive minds are not a common place in current robot technology, science fiction is already full of them: the "Borg" in *Star Trek* or the "machines" in the *Matrix* are prototypical examples, but even the OS in *Her* is really a hive mind (able to interact with many humans at the same time and able to fully share any information it has gained about humans from its many parallel dialogue interactions). In a hive mind, every individual agent (or "agent body", rather) knows what every other agent (or agent body) knows. Individual agents do not need to explicitly communicate to share information, or to learn about each other's tasks and goals as their knowledge is always perfectly synchronized. While there are several definitions of "hive mind", e.g., the collective mental activity expressed in the complex, coordinated behavior of a colony of social insects (such as bees or ants) regarded as comparable to a single mind controlling the behavior of an individual organism" (e.g., [3]), we will use the term in the sense of "collective thoughts, ideas, and opinions of a group of [agents] regarded as functioning together as a single mind".

Note that we already have various types of such hive minds, but admittedly not very interesting ones. Cloud-based chatbots and natural-language enabled agents like Alexa, Siri, Cortana, Google duplex, are hive minds. They are available for different devices and typically only function (or function well) when connected to their cloud-based database and analysis algorithms. We do no re-

ally construe the different instances of these agents as being *different* in kind (e.g., "my Alexa" vs. yours). We understand that in some sense they are all the same agent (they do not form memories of particular interactions or knowledge exchanged, at least not yet). Amazon's warehouses are another example of a hive mind where human packers work with the robot controlled by a central algorithm in charge of optimizing robot behavior such as item delivery. Different from virtual agents, Amazon warehouses are really examples of a hive mind with many physical bodies controlled by one mind, closer resembling the concept of the Borg (with the central controller being akin to the "Borg queen"). However, analogous to virtual agents, people do not view these robots as individuals, and consequently do not track theme and their performance, and interact with them outside of grabbing items they deliver ("Hey, Yellow 5, you are always late in bringing me supplies...").

Overall, hive minds seem to be promising solutions for a variety of robotic contexts beyond warehouses and cloud-based virtual agent, e.g., in mixed initiative human-robot teams, because they can build and maintain truly shared mental models which they can then use to support their human teammates in unprecedented ways. The critical difference between human and robot shared mental models is that robots can truly share them, by way of sharing some of the components in their agent architecture (i.e., the same component is part of multiple architecture instances). As such, in a hive mind setting, individual robots do not have to synchronize their individual mental models through external communication (as humans must do), because they have a *shared mental model* of the world. Moreover, robots can automatically include any mental and perceptual state of any other robot in the hive to improve their own behavior.

3 The Mind-Sharing DIARC Architecture

The DIARC is a component-based cognitive robotic architecture [5, 4] that is a hybrid of a cognitive and a robotic architecture implemented in the *Agent Development Environment* (ADE) [6]. It includes robotic capabilities for perceptual and action processing, as well as motion and task planning, but also cognitive capabilities for natural language understanding, reasoning, and instruction-based learning. Different from other cognitive architectures, it is intrinsically component-based where each component operates asynchronously with other components (very much in the style of typical implementations of robotic control architecture in distributed middleware like ROS [2] and others). Different from other robotic architectures, however, it has various short and long-term memories to store knowledge about objects, actions, words, discourse, interactions, tasks, and episodic traces, all indexed where applicable by explicit references to agents in a group or "hive" for which the item applies. Most importantly, analogous to starting services in robotic middleware like ROS, DIARC can be configured with different components present based on the application demands and components can be added or removed at run-time. For multi-agent settings where multiple instances of the same component are available in the system,

DIARC, different other multi-robot configurations (e.g., a set of ROS services, say), provides mechanisms for explicit tracking which component belongs to which agent. For example, the architectures of two robots might both require an instance of a vision component and the specific component connected to the requisite robot's camera is marked by a robot ID in the configuration which can then be used in the system to route services and their consumers. For example, if the motion planner on Robot1 needs to access the camera on Robot1, it can request a "look-for" action with the constraint that the service providing component be part of the "group Robot1". On the other, if Robot1 needs to get visual information from Robot2, it would not have to request it explicitly, but could just perform the "look-for" action with constraint "group Robot2", thus performing the action on Robot2's camera.

By explicitly exposing these identifiers to components in the architecture, they become accessible to introspection and available for reasoning which has many advantages compared to approaches that implicitly assign components to robots and hide that information in the middleware. For example, they enable accessing device and state information on different robots easy from any robot in the shared system, allowing a reasoning component, for example, to determine what capabilities are available in the system and how to utilize them. The shared information also becomes accessible to the natural language subsystem, thus allow users to explicitly address different robots, utilizing their joint perceptions, and actuation capabilities, and overall treat the multi-robot system as a "hive mind" with different body instances.

To illustrate the potential of component-sharing for multi-robot instructions and tasks, we will use the architecture depicted at the bottom of in Fig. 1 for demonstrations of (1) for (1) multi-robot instructions, (2) automatic information gathering and sharing utilizing capabilities and information of other robots, and (3) multi-robot knowledge acquisition (for the last demonstration, we also include a third robot, see below). In this configuration, the two robots share "everything they can possibly share in their architecture" except for the components representing their bodies (sensors and actuators) which need to be separate. By sharing the rest of the architecture, it is possible to access sensor states (by instructing sensing actions) and initiating behaviors (by instruction actions) on any of the platforms mediated by any of the robots. And the setting is not limited to two robots, but works for any number of robots; and it is not limited to homogeneous robots either, but equally works for heterogeneous robots (see the third example below).

3.1 Instructing Hive Minds

Consider a setting with two Nao robots, Dempster and Shafer, were a human instructor, Brad, just issued the command "Dempster, tell Shafer to stand" to the Dempster robot (see [1]). After speech recognition (ASR in Fig. 1), the natural language understanding (ALU) system produced the semantic representation

```
want(Brad,Dempster,do(tell(Dempster,Shafer,do(stand(Shafer)))))
```

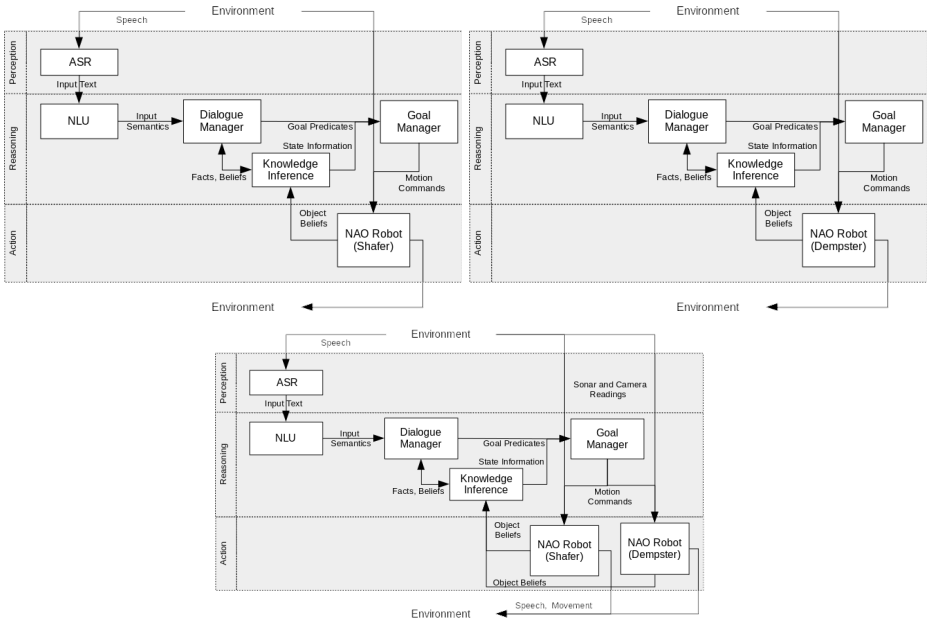

Fig. 1. The individual agent architecture instances for the Shafer and Dempster robots (top) vs. the component-sharing architecture instance for both robots (bottom), see text for details.

which gets passed on to the shared Dialogue Manager (DM) and causes it generate the following goal to be submitted to the shared Goal Manager (GM) for Dempster:

```
tell(Dempster,Shafer,do(stand(Shafer)))
```

Being of the form "tell(X,Y,do(Z))" and since Y is Shafer, an agent in the system, the DM can directly generate another goal, this time for Shafer, to do Z, which also gets submitted to the shared GM:

```
stand(Shafer)
```

It is this second goal that then causes the "stand action" to be executed on the Shafer robot, thus making the Shafer robot stand up (see `http://hrilab.tufts.edu/movies/2bots1brain.mp4` While the robots are shown in the same space in this video, they could be in separate spaces as long as they can connect to the same underlying network that allows the ADE middleware to share components (e.g., they could be in different rooms, different countries, etc.). Hence, this setting is different from one where one robot "overhears" what the other is being told, but processes that instruction itself, and generates explicit representation of the goals and knowledge of the other robot. It is also worth pointing out that nothing in the system hinges on giving robot bodies individual names

or treating robot bodies as individuals, the instructor could have just as easily said "Robot, tell body 1 to stand", "Robot, does body 2 see an obstacle", etc.

Similarly, for the question "Does Shafer see an obstacle?" the semantic representation is

```
check(Dempster,check(Shafer,see(Shafer,obstacle)))
```

which causes the execution of the "check(X,Y)" script which makes agent X attempt to verify Y, in this case another check goal to be submitted to the shared GM for Dempster:

```
check(Shafer,see(Shafer,obstacle))
```

This causes the check script execution causes the execution of primitive predicate "see(X,Y)" for Shafer

```
see(Shafer,obstacles)
```

which returns true if Shafer sees an obstacle, which it does, and hence the affirmative response by Dempster.

3.2 Automatic Information Sharing in Hive Minds

With hive minds, the level of sharing is critical for understanding what they can or cannot do automatically. For example, robots that only share their knowledge base might not be able to answer the kinds of perceptual questions asked in the example above because they would have to explicitly initiate a perception action on one of their bodies to get that information because it is not automatically available, and they might not be able to trigger such perception actions (unless they are connected in other ways and have explicit control mechanisms that allow them to make such requests). As a result, it is important for human interactants to understand the level of integration and sharing of hive minds in order to have reasonable expectations about their capabilities and behaviors.

For example, hive agents can automatically use the presence of other hive agents to augment their perception without the need for explicit instruction (which, without knowing, can be surprising if not plainly eerie for humans, see the Discussion section).

Consider a setting in which the Dempster robot is situated on a table and a human instructor, Ravenna, is engaging the robot in the following dialogue:

```
R: Hello Dempster.
D: Hello Ravenna.
R: Please stand.
D: OK.
R: Walk backward.
D: I cannot walk back because I do not have rear sensors.
```

Ravenna then picks up the Shafer robot and puts is behind Dempster on the table (the settings is depicted in Fig. 2).

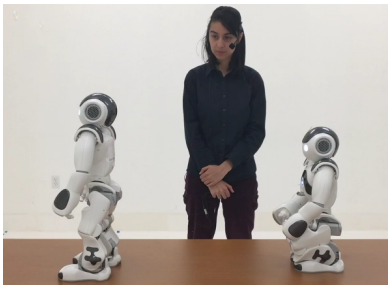

Fig. 2. The instruction setting allow the Dempster robot (left) to automatically infer that the area behind it is safe based on the perceptions of the Shafer robot (right) which are automatically shared.

```
R: Can you walk backward now?
D: Yes.
```

Dempster starts walking backwards. This is possible because Shafer, after having been placed behind Dempster, recognized Dempster and determined that there was open space in front of it, hence Dempster could reason that there was open space behind it given that it was located in front of Dempster. This reasoning requires several facts about locations and relations among located items (we use "ifo" as an abbreviation for "in front of"):

$$see(x,y) \rightarrow isLocated(x,behind,y)$$
$$see(x,y) \rightarrow isLocated(y,ifo,x)$$

and also about the relation between "ifo" and "behind":

$$isLocated(x,behind,y) \rightarrow isLocated(y,ifo,x)$$
$$isLocated(x,ifo,y) \rightarrow isLocated(y,behind,x)$$

Moreover, open space in front of a robot needs to be related to not seeing any obstacle and to open space behind the robot:

$$\neg see(x,o) \wedge obstacle(o) \rightarrow openspace(ifo(x))$$
$$openspace(ifo(x)) \wedge isLocated(x,behind,y) \rightarrow openspace(behind(y))$$

Equipped with the above principles, because Shafer can see Dempster, "isLocated(Dempster,ifo,Shafer)" and "isLocated(Shafer,behind,Dempster)" are derivable. And because Shafer does not see an obstacle in front, the additional "openspace(ifo(Shafer))" and "openspace(behind(Dempster))" are derivable. This information can then be used in conjunction with the pre-condition to moving in direction (forward or backward) that the respective area (in front or in the back) is safe to initiate the robot motion. Once Shafer detects that there is no longer open space in front, Dempster also detects that there is no open space behind it and thus stops walking (see `https://hrilab.tufts.edu/movies/naospatialinference.mp4`).

```
D: I cannot move back because I do not know that the area behind me is
   safe.
```

It is possible for the human instructor to find out what the Shafer robot knows (i.e., all the facts the system automatically inferred based on Shafer's perception of Dempster), and hence what information the Dempster robot has access to:

```
R: Hello Shafer.
S: Hello Ravenna.
R: Where is Dempster?
S: Dempster is located in front of me.
R: Do you see an obstacle?
S: No.
R: Do you think the area behind Dempster is safe?
S: Yes.
R: Walk backward Dempster?
D: OK.
```

Again, Dempster starts walking backwards and stops as before.

```
D: I cannot move back because I do not know that the area behind me is
   safe.
R: Where is Shafer?
D: Shafer is located behind me.
R: Is there open space behind you?
D: No.
```

4 Discussion

Hive minds pose interesting challenges for natural language interactions, not only from an architectural point of view of how to represent natural language semantics and to store facts that concern multiple agents effectively, but also from a human interaction and dialogue design perspective. For talking to "many that are one" is not an easy feat for humans, partly due to how we conceptualize agents: Who do you address in a hive? And who do you think you are talking to? How do you know that when you talk to one agent that all other agents will know it too or what they will know? For example, will agents know who you talked to and does this even matter? Humans keep episodic traces of important dialogues that include information about the interactants (their identity, what they said, etc.), how would this be different in a hive where you might need to talk to multiple agents that you treat at some level as individuals when you talk to them, but that at another level are essentially one?

We currently do not have answers to the above questions, but our own past work has attempted to answer the question of how people react when robots share everything and in way that is not transparent to them. Specifically, we investigated the question whether robots' tacit communication (either by exchanging messages or by sharing a mind) would be acceptable to humans in contexts where they had to instruct a robot to relay task-based information to another robot [7]. After the first robot had received the human instructions, subjects observed the first robot driving up to the second robot (which was located

in a different room) and relaying the human instructions to the second robot in spoken natural language in one experimental condition, while in other experimental condition the second robot had already started to perform the instructed task when subjects entered the area and were able to observe it. Hence, subjects were led to believe that the robots tacitly communicated with each other. Overall, we found that human instructors found tacit or covert communication creepy or unsettling – they wanted to be "in" on what was going on. This raises the important question of when and how hive minds should communicate with humans to make their behaviors and interactions as intuitive and acceptable to humans as possible, e.g., when should robots verbalize shared information and explicitly point it out, or how far could robots go with initiating team-relevant actions based on their joint knowledge without confusing humans. It may well be that under time pressure humans will care less about overt communication and may even prefer covert information sharing, but settling this issue will require further experimental work.

Another interesting aspect about hive-based communication concerns potential inconsistencies that might arise as a result of how humans instruct agents in conjunction with the knowledge sharing and overall operation of the hive mind. Consider the instruction "Shafer, do not tell Dempster that the area behind you is safe" – What is Shafer supposed to do? It could not (explicitly) tell Dempster this fact, which is what would currently happen in the component-sharing architecture anyway, i.e., the "tell" action would not be executed, and hence there would not be a direct assertion of the fact that the area behind Shafer is safe resulting from a "tell" action. However, the fact would nevertheless be asserted by Shafer based on the natural language semantics obtained from processing of the instruction and certain reasoning principles that Shafer automatically applies (e.g., that in a cooperative setting one is never asked to tell another team member a false proposition, and that one would never ask a team member to not tell a false proposition, because the team convention is to only tell truths). Hence, if Shafer knows that the area behind it is safe, then Dempster knows it too, for they both share the same knowledge. But then pragmatically saying "OK" (as the robot would in response as it truthfully did not "tell Dempster") is misleading at best given that Dempster already knows. To prevent Dempster from knowing, Shafer would have to not assert the fact, which might work in the above case, but not with the instruction "Remember that the area behind you is safe, but do not tell Dempster that". Here, Shafer would have to point out the conflict that it cannot do both, and that would require it to employ some meta-reasoning taking the architectural configuration into account, which it is not capable of performing at present. Similar examples can be constructed that ultimately will require hive minds to have an elevated level of awareness of what different participating bodies might mean for people and how shared components enabling shared perceptions, knowledge, and actions can interfere with the natural human assumption and understanding of hive-mind bodies as individual agents.

5 Conclusion

Hive minds open up promising opportunities for improving coordination in mixed-initiative human-robot teams and thus achieving higher task performance, as mind-sharing agents naturally implement *shared mental models* (that are the basis of human teamwork) and do so more efficiently and better than humans ever could. However, task-based natural language interactions with hive-minds can be challenging for both human interlocutors and for robot architecture developers as they have to negotiate the tension between individual agent bodies and the hive mind which can seem contradictory from a human perspective. It is clear that as hive-mind technologies are being developed further, we need to better understand the various tradeoffs and human preferences, which will required extensive experimentation in settings with multiple mind-sharing robots, be they realized via component-sharing or some other technology. In particular, it will be necessary to investigate the potential language challenges posed by hive for humans and how they can be address for us to be able to fully utilize the potential of hive-minds without negative effects.

Acknowledgment

This work was in part supported by AFOSR grant FA9550-18-1-0465. Thanks also to the members of the Tufts HRILab for their support and help.

References

1. Oosterveld, B., Brusatin, L., Scheutz, M.: Two bots, one brain: Component sharing in cognitive robotic architectures. In: Proceedings of 12th ACM/IEEE International Conference on Human-Robot Interaction Video Contest (2017)
2. Quigley, M., Conley, K., Gerkey, B.P., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: Ros: an open-source robot operating system. In: Proceedings of ICRA Workshop on Open Source Software (2009)
3. Reynolds, C.: Flocks, herds and schools: A distributed behavioral model. In: SIGGRAPH '87: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques. pp. 25—34. Association for Computing Machinery (1987)
4. Schermerhorn, P., Kramer, J., Brick, T., Anderson, D., Dingler, A., Scheutz, M.: Diarc: A testbed for natural human-robot interactions. In: Proceedings of AAAI 2006 Mobile Robot Workshop (2006)
5. Scheutz, M., Williams, T., Krause, E., Oosterveld, B., Sarathy, V., Frasca, T.: An overview of the distributed integrated cognition affect and reflection diarc architecture. In: Ferreira, M.A., Sequeira, J.S., Ventura, R. (eds.) Cognitive Architectures, Intelligent Systems, Control and Automation: Science and Engineering, vol. 94. Springer (2019)
6. Scheutz, M.: ADE: Steps toward a distributed development and runtime environment for complex robotic agent architectures. Applied Artificial Intelligence **20**(2-4), 275–304 (2006)
7. Williams, T., Briggs, P., Scheutz, M.: Covert robot-robot communication: Human perceptions and implications for hri. Journal of Human-Robot Interaction **4**(2), 23–49 (2015)