12th ICCRTS

Adapting C2 to the 21st Century

Human and Software Factors for Successful System Adaptation

C2 Metrics and Assessment; Cognitive and Social Issues; C2 Technologies and Systems

Douglas S. Lange, SPAWARSYSCEN San Diego
Michael Carlin, SPAWARSYSCEN San Diego
Volodymyr Ivanchenko, Naval Postgraduate School
Luqi, Naval Postgraduate School
Valdis Berzins, Naval Postgraduate School

POC: Douglas S. Lange
SPAWARSYSCEN
53560 Hull Street
San Diego, CA 92152-5001
619-553-6534
doug.lange@navy.mil

ABSTRACT

Machine learning is an important tool towards adaptive command and control. The Defense Advanced Research Projects Agency (DARPA) has implemented a program to build the first instance of a complete cognitive agent that uses machine learning. The program, called Personalized Assistant that Learns (PAL), is expected to yield new cognitive technology of significant value not only to the military, but also to business and academic sectors.

PAL uses a variety of learning algorithms to allow assistants to adapt to the operational needs of its human user. The PAL Boot Camp concept was created to address the training of these assistants to ensure that they are useful when they are deployed and that they will successfully learn to perform tasks.

As part of the Boot Camp experimentation, a simulation was created to determine what aspects were important in ensuring that assistants would enhance the performance of their users. The results of experimentation with the simulation show the relative importance of the methods of human use, observability, and patterns of initial training. This paper reports on these results.

Human and Software Factors for Successful System Adaptation

Douglas S. Lange, SPAWARSYSCEN San Diego
Michael Carlin, SPAWARSYSCEN San Diego
Volodymyr Ivanchenko, Naval Postgraduate School
Luqi, Naval Postgraduate School
Valdis Berzins, Naval Postgraduate School

ABSTRACT

Machine learning is an important tool towards adaptive command and control. The Defense Advanced Research Projects Agency (DARPA) has implemented a program to build the first instance of a complete cognitive agent that uses machine learning. The program, called Personalized Assistant that Learns (PAL), is expected to yield new cognitive technology of significant value not only to the military, but also to business and academic sectors.

PAL uses a variety of learning algorithms to allow assistants to adapt to the operational needs of its human user. The PAL Boot Camp concept was created to address the training of these assistants to ensure that they are useful when they are deployed and that they will successfully learn to perform tasks.

As part of the Boot Camp experimentation, a simulation was created to determine what aspects were important in ensuring that assistants would enhance the performance of their users. The results of experimentation with the simulation show the relative importance of the methods of human use, observability, and patterns of initial training. This paper reports on these results.

INTRODUCTION

The Defense Advanced Research Projects Agency's (DARPA) Personalized Assistant that Learns (PAL) program is an attempt to develop a class of cognitive systems that serve as assistants. What sets this program apart is the emphasis on the integration of a variety of machine learning techniques in order to allow learning of new tasks after the capability is deployed (Brachman 2007).

The U.S. Navy and presumably other organizations responsible for developing and deploying mission critical systems, uses a structured engineering process for deciding when a new capability is ready for deployment. Systems must pass a cost versus benefit analysis, where one measures benefit by the ability to meet documented requirements. Installing software that is going to learn to meet user requirements only after being deployed for some unspecified time is problematic.

Software and system engineers will need a new way to analyze capabilities that will learn after deployment. Likewise, we will need to learn how to prepare such systems and their users to ensure success.

POSSIBLE OUTCOMES

The PAL program is developing software to be used as part of a human-computer team. This is natural in the decision support area, and we can use metrics relevant to such systems.

Figure 1 illustrates notional performance curves for the human-computer team for each of three scenarios that we will call expected, abandonment, and disaster. From the introduction of a PAL at point 'A', we see a natural degradation of performance initially. We might see a bottoming or at least slowing of this cost by the time labeled 'B'. If the user sees improvement early on, we may be on the expected curve, and performance could continue to improve and overtake previous levels of performance at point 'E'. Learning tends to plateau at some point, which for the graph below we label as 'F'. If performance does not begin to improve quickly enough, the user is likely to abandon the attempt to use the assistant and resume performing the task alone. The point 'D' shows performance resuming pre-deployment levels after the assistant is abandoned. The final scenario is that while performance never improves, the user continues to try to use the assistant and performance continues to deteriorate. This is represented by the disaster curve in figure 1.

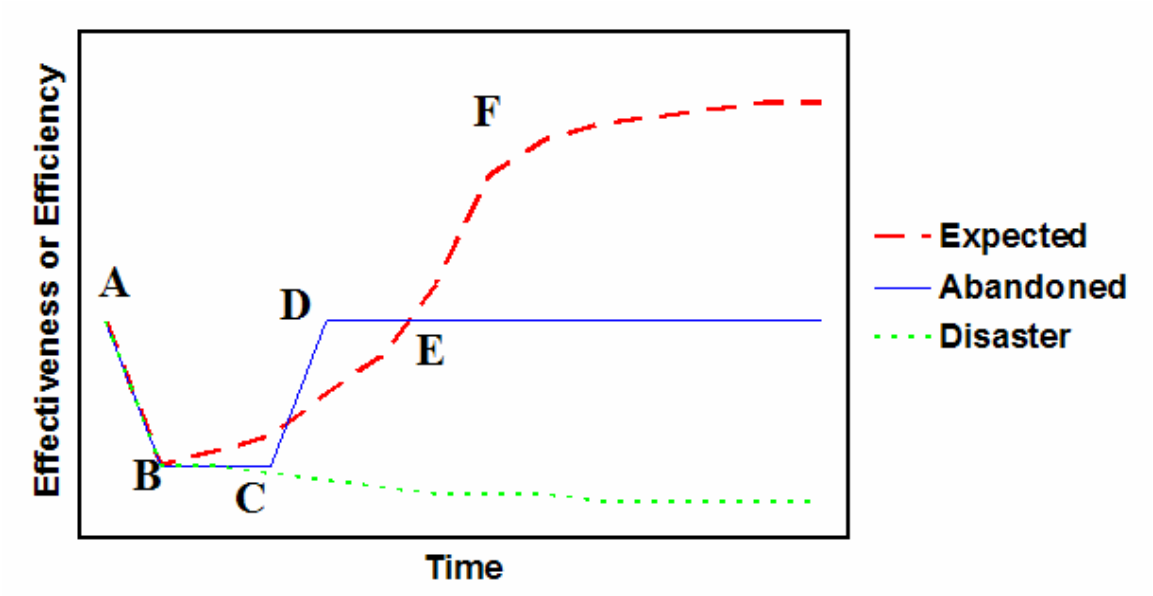

Figure 1 Notional Performance Curves (Lange 2007)

GOALS OF THE BOOT CAMP

Anthropomorphic descriptions are difficult to avoid when thinking about the PAL program. The inspiration behind the program, quite literally, is benefit gained by those fortunate enough to have a human assistant. There are however, differences that immediately come to mind when one delves deeper into the possible capabilities of a PAL when compared with a human assistant. An example is that two or more PAL should be able to transfer knowledge among one another, rather than first describing that knowledge in some language that can only provide an approximation of the needed information with the hope that the receiver correctly interprets the message.

However, there are many similarities between a PAL and a human assistant. For example, both must learn how their boss wants to conduct business, their performance is best measured by the change in their boss's performance, and if not found useful will get cut out of processes where the boss feels that it is more expedient to just do it him or herself. Because of the many similarities, it is

useful to consider how the military deals with the introduction of humans into the operational environment and decide if similar processes are useful for PAL.

Structured training is one technique used by the military to prepare humans for their jobs. Most famous is basic training, also known as boot camp. New recruits in the military already have been to school (e.g., high school), but need to learn the basics of the military domain. For a PAL, it may be easier to have it learn from experiencing the military domain within a training setting rather than having an engineer decide how to program it.

Another technique used in the military for training humans is simulation. Officers moving to a joint staff tour are taught crisis action planning (CAP) processes in training driven by simulation systems. To be successful, the officers need to have certain amounts of prior knowledge. Officers are tested in their knowledge before leaving. There is no thought that the real life situations will exactly mirror those they have encountered in training. Rather it is believed that the skills they gain will allow them to learn the rest of what they need to know while on-the-job.

This approach parallels the goals of the in-the-wild learning by PAL. The assistant must learn within its operational environment, but in order to do so, must have some basic knowledge about its domain and the basic processes that are likely to be encountered.

BOOT CAMP MODEL

Our approach has three phases, with the transition points offering opportunities for measurement. The phases are 1) pre-boot camp programming of the knowledge base, 2) boot camp training, and 3) operational use (Lange 2006a).

In the pre-boot camp phase, we add knowledge through programming. This can be either using traditional programming languages or as statements in a logic-based knowledge base, or by any number of means. The distinction is that the method is one of designed and engineered programming. Learning algorithms may be employed, but through a controlled method, and through training data sets.

BOOT CAMP FOR COMMAND AND CONTROL

Prior to entry into the boot camp there is an opportunity to measure the knowledge currently held in the assistant. In the PAL program, an adaptation of traditional standardized tests for human students is used (Cohen and Pool 2005). PAL is meant to interact with human users, so using test techniques that are applied to humans seems appropriate.

The consequences of inadequate programming are that the assistant will not function well enough to learn in the simulation environment of the boot camp. This is the same issue we are hoping to avoid in operational use. Therefore, the measurements that support the decision process are equally valid in both environments. It will be a simpler matter to measure performance changes in the boot camp environment than in the operational environment and if we can use it to approximate the operational environment, we will be well off.

The next phase, the boot camp, is characterized by the use of simulation systems, just as they are used for training humans. We intend to use the Joint Semi-Automated Forces (JSAF) and the Composeable FORCEnet (CFn) command and control capability to provide a simulated operations environment for the assistant to observe and participate in. JSAF can be used to pose problems for

decision makers to respond to. CFn allows the users to collaborate with PAL observing, participating, and learning. Decisions are enacted within the JSAF environment and results observed.

The questions in devising this boot camp capability are: how much training needs to be provided, and how broad and how deep does the resulting knowledge need to be at the end of the training to suggest that successful deployment will result? The remainder of this paper discusses an exploration of these questions.

BOOT CAMP SIMULATION FOR EXPERIMENTATION

In preparation for developing training plans for use with humans and PAL software, we created a simulation of the boot camp in order to determine the patterns of learning that were needed. The simulation incorporated models of the task environment, the assistants, operations, and models of the human behavior for using the cognitive systems (Lange 2006b).

We model the environment using the tasks and concepts available to be known and performed. This approach uses a model that is commonly used in agent scheduling research (Chandrasekaran 1992) and was close to what was used in the PAL program for task learning (Meyers 2006). Figure 2 illustrates the basic structure using Unified Modeling Language (UML).

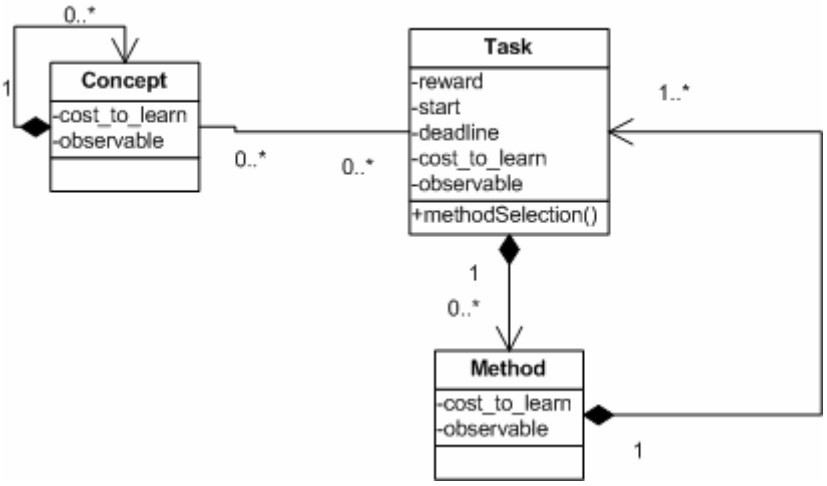

Figure 2 Task Structure in UML (Lange 2007)

Tasks and methods form bipartite trees, where tasks are made of methods, and methods are composed of tasks. One executes all tasks of a method, but only needs one method to perform a task. Tasks at the leaf nodes are associated with concepts from a hierarchy of concepts. This is an implementation of the Task-Method-Knowledge (TMK) approach (Murdock 2000). The association represents that a concept is needed to be known in order to perform the task, in addition to knowing the steps of the task. A forest of task-method trees, associated with a concept tree form an environment model. Tasks, methods, and concepts are marked with whether or not they are observable, and with the cost to perform or learn the task, method, or concept. Various rules are required for the semantics of these objects that are described in (Lange 2007).

The knowledge held by an assistant is modeled as a marked environment model, where the marks indicate which elements of the model the assistant has learned. Operations are modeled as lists of instances of tasks. The tasks are taken from the roots of the task trees. Each instance has a deadline and a reward value for finishing the task before the deadline. Task instances are also associated with a small number concepts from the concept tree. These concepts are independent of the concepts that

the tasks are already associated with from the TMK model. When deployed, PAL will face this situation frequently. A task that researches the casualty status of a piece of shipboard equipment requires knowledge of how to research the information, but may also need some knowledge of the particular ship involved.

The human is modeled in the execution of the simulation. We are interested in the decisions made by the human as to what method to use when performing a task. The human must decide when to expend effort to provide training for the assistant and when to perform the task without help. Any task that the assistant can perform without additional training costs the human no time. Training the assistant to perform a task costs some factor more than the time it would take the human to perform the task alone. Training the assistant to know concepts related to a task has a cost associated as well.

EXPERIMENT DESIGN

PAL aims to use multiple learning strategies to enable an assistant to be successfully utilized. The experiments using the simulation were designed to help identify the roles and requirements for different learning strategies and the level of preparation the assistant needed prior to deployment to take advantage of them. In order to accomplish that, experiments were run against ten different human behavior algorithms with variations in the complexity of the environment, the amount and type of preparation of the assistant and the ability of the assistant to learn by observation rather than more costly training. Table 1 shows the ways the experiments differed.

Number of Operations	Observability	Max Depth	Coverage
5	10%	5	0%
5	10%	5	10%
5	10%	5	20%
5	10%	5	30%
5	10%	5	100%
5	10%	5	10% focused
5	10%	5	10% focused 10% additional
5	10%	5	10% focused 20% additional
5	10%	5	10% focused 30% additional
5	10%	3	0%
5	10%	3	10%
5	10%	3	20%
5	10%	3	30%
5	10%	3	100%
5	10%	3	10% focused
5	10%	3	10% focused 10% additional
5	10%	3	10% focused 20% additional

Number of Operations	Observability	Max Depth	Coverage
5	10%	3	10% focused 30% additional
5	10%	9	0%
5	10%	9	10%
5	10%	9	20%
5	10%	9	30%
5	10%	9	100%
5	10%	9	10% focused
5	10%	9	10% focused 10% additional
5	10%	9	10% focused 20% additional
5	10%	9	10% focused 30% additional
5	0%	5	20%
5	0%	5	10% focused 10% additional
5	10%	5	20%
5	10%	5	10% focused 10% additional
5	20%	5	20%
5	20%	5	10% focused 10% additional
5	30%	5	20%
5	30%	5	10% focused 10% additional
5	100%	5	20%
5	100%	5	10% focused 10% additional

Table 1 Experimental Operations

Parameters and fixed environment generation settings were chosen in part by observations made in an earlier set of experiments in which Crisis Action Planning processes were executed by military personnel as part of a role-playing simulation (Wong et al. 2006). Training in the boot camp consisted of either random training, or a combination of focused and random training. Focused training provided that a small number of tasks would be completely known to the assistant. Maximum depth of the concept tree provides differing levels of complexity for the operational environment being modeled, and observability indicates the amount of the environment that can be learned by the assistant without explicit help from the human.

Five operations of 100 tasks were generated for each set of experiments. Each operation was repeated for ten different human behavior algorithms (see Table 2), and data was collected on the effectiveness (reward collected) and efficiency (reward collected divided by time used). Analysis of variance was then used to determine the contribution of each of the variables towards the results. Graphical analysis was also performed to better understand the results.

Human-only Greedy-Fast	Human performs the task alone, using most efficient method.
Greedy-Fast	Human chooses method that with assistant will be most efficient, without being willing to train the assistant during the operation.
Greedy-Enlightened	Similar to greedy-fast, but the human will consider methods that require task-instance dependent concept training for the assistant if it will result in a faster completion.
Realistic #1	First choice will be the fastest method that involves teaching task instance related concepts so that the assistant's knowledge is built up, but that will be fast enough even with teaching to meet the deadline.
Realistic #2	Similar to Realistic #1, but consider teaching new tasks and their concepts as well as task instance related concepts.
Realistic #3	Same as Realistic #2, but now if there is any spare time built up, use it to further train the assistant.
Human-Favorite	The human performs the task alone using one path through the TMK tree that represents a favorite way to perform the task.
Human Favorite with Assistance	Same as Human Favorite, but now the assistant can help if it knows any of the tasks on the favorite path and knows the associated task instance related concepts. Teaching is in accordance with Realistic #3
Smart Human Favorite	Same as Human Favorite, but if the favorite method is not fast enough to earn a reward, the next time this task is performed a new method will be tried.
Smart Human Favorite with Assistance	Same as Smart Human Favorite, but now the assistant can help if tasks and concepts are known. Teaching is in accordance with Realistic #3.

Table 2 Human Behavior Algorithms

EXPERIMENTAL RESULTS

The results showed that the environment complexity had little to do with changes in performance relative to other factors. While statistical significance was found in the data, the relative contribution to the change was small. The F test was run for each variable as part of an Analysis of Variance (ANOVA) and was significant for the tree shape factor.

$$F(2,747) = 4.110,\ p < 0.02$$

However, the proportion of the total variance that attributed to this effect is low. Partial Eta Squared is 0.01. The interpretation of this result relative to the results that will be in the sections on assistant training and human behavior is that it is of less importance than how the assistant is trained in the boot camp and the manner in which the assistant is employed and trained by its user.

Observability appears to contribute positively to the effectiveness of the assistant. However, it does not appear to contribute much until the observable percentage gets very high. ANOVA showed the following main effects of observability.

$$F(4, 495) = 123.1\ ,\ p < 0.01,$$

Partial Eta Squared = 0.53

Further analysis shows that observability above 30% was significantly better than if it was below 30%. The graphs below in Figure 3, show the effectiveness at varying levels of observability. These graphs are smoothed using a sliding window that is calculated in the following way. Points are calculated as an average effectiveness over the last *w* (for window size) operations. For the first *w-1* points, the data is padded with human only effectiveness numbers. In this way, we are actually starting our analysis *w* operations before the start of the test when the assistant is introduced. The calculations are as follows for the first *w-1* points:

$$\frac{(w-i)average\_achieved\_award\_for\_base+\sum_{j=1}^{i}reward\_achieved_j}{(w-i)average\_available\_award\_for\_base+\sum_{j=1}^{i}reward\_available_j}$$

Starting with point *w*, the calculation is as follows:

$$\frac{\sum_{j=i-w}^{i}reward\_achieved_j}{\sum_{j=i-w}^{i}reward\_available_j}$$

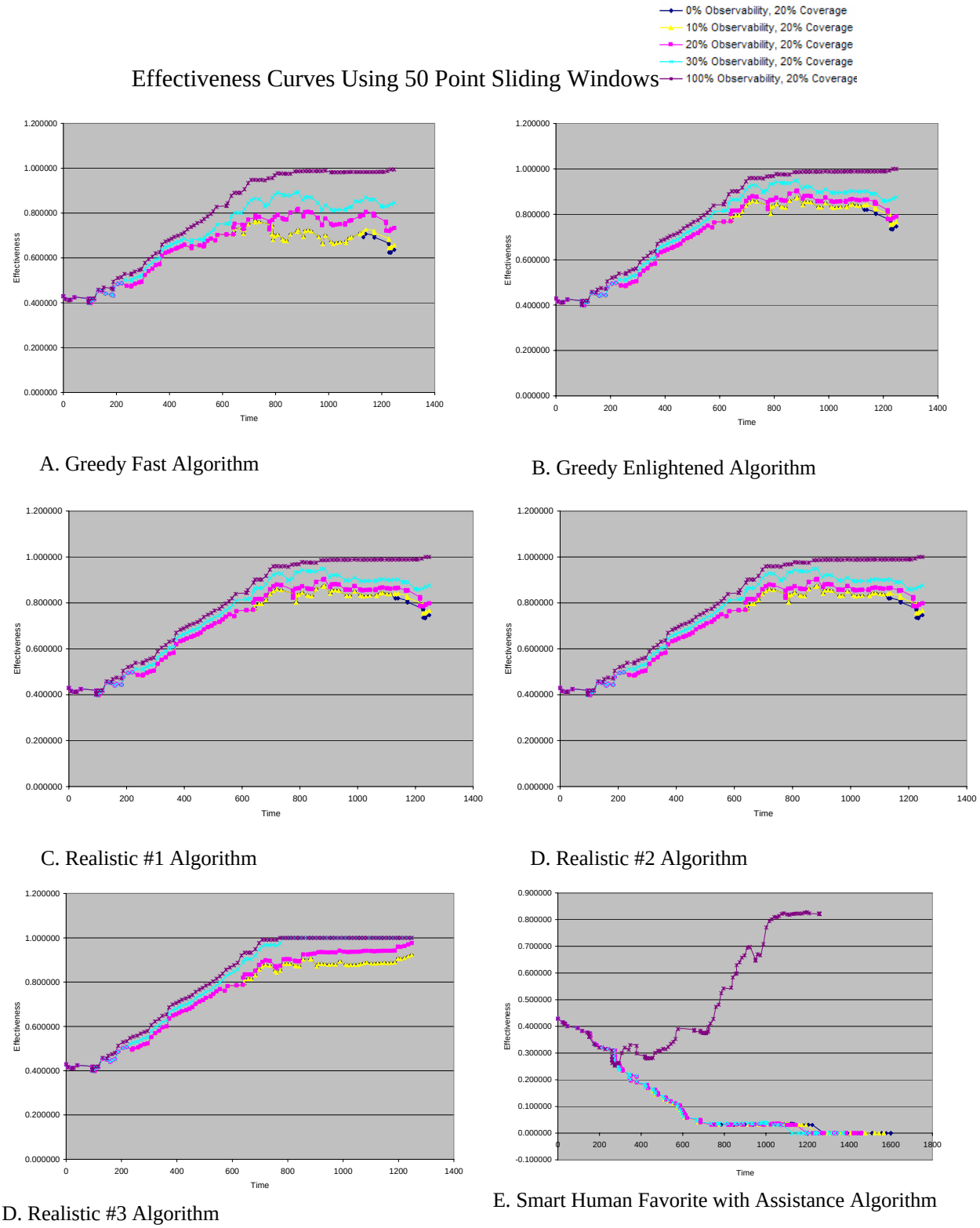

Figure 3 Effectiveness at Different Levels of Observability

Analysis varying the amount of knowledge that the assistant gained in the boot camp yielded significant results. Tests were performed with 0%, 10%, 20%, 30%, and 100% knowledge of the task-method and concept trees. ANOVA was performed to see if the factor was significant in determining effectiveness of the human-computer team. The results showed that not only was this a significant factor, but that it accounted for a large portion of the variation seen in the experiments.

$$F(4, 745) = 137.7, p < 0.01$$

Partial Eta Squared = 0.429

The quantity of the knowledge is an important factor. The question suggested is how much is needed? The experiments here can only answer definitively for the particular environments and operations used, but can suggest patterns that can inform boot camp development. In order to get those patterns, we turned to time series data from the experiments. Graphical analysis suggests that for the weaker human behavior strategies, there is a threshold value somewhere in the 20-30% range where an assistant can avoid abandonment.

Results for the type of training provided (random or manipulated) were ambiguous. The Realistic #3 algorithm curves demonstrated most clearly what is happening. There are situations where the assistant with 20% manipulated knowledge, performs better than the assistant with 30% manipulated knowledge. Since the two assistants can know about a different set of tasks, there is always a chance that the 20% manipulated assistant just happens to know the tasks that appeared early in the operation. What is also apparent is that if an assistant is unlucky enough not to know the early tasks, then having broad knowledge is an advantage, since it saves time that allows for teaching. This implies that if a boot camp is to be successful, the tasks selected for the training the assistant must be chosen carefully to be the ones most likely seen early in its employment. If that is not the case, then random instruction may do better.

Statistical analysis with ANOVA showed that the human behavior algorithm was a significant and powerful factor in the results.

$$F(9, 740) = 256.2, p < 0.01$$

Partial Eta Squared = 0.759

The Realistic #3 algorithm is the superior approach (based on graphical analysis) for employing the assistant. There are also instances where Realistic #2 outperforms the others, but no result is more obvious than the benefits of using Realistic #3 and this clearly must explain the power demonstrated in the ANOVA results.

The primary difference in Realistic #3 is the addition of spare-time training. Each of the algorithms builds on the previous one adding opportunities for training, but only this addition causes such a large leap in results.

The human-favorite series of algorithms were dismal failures. This was regardless of the amount of or type of training provide to the assistant in the simulated boot camp. Only the assistant with the manipulated learning with an additional 30% coverage of random learning showed promise. It would likely avoid abandonment with a patient user as it in general kept the user’s performance above the baseline of the human-only favorite effectiveness. The fault lies not with the assistant in this case. This is a failed human strategy for getting work done, regardless of the assistant. These experiments show that with enough learning, an assistant might be able to bail out a poor human strategy. Figure 4 below, shows the effectiveness curves for the different human behavior algorithms.

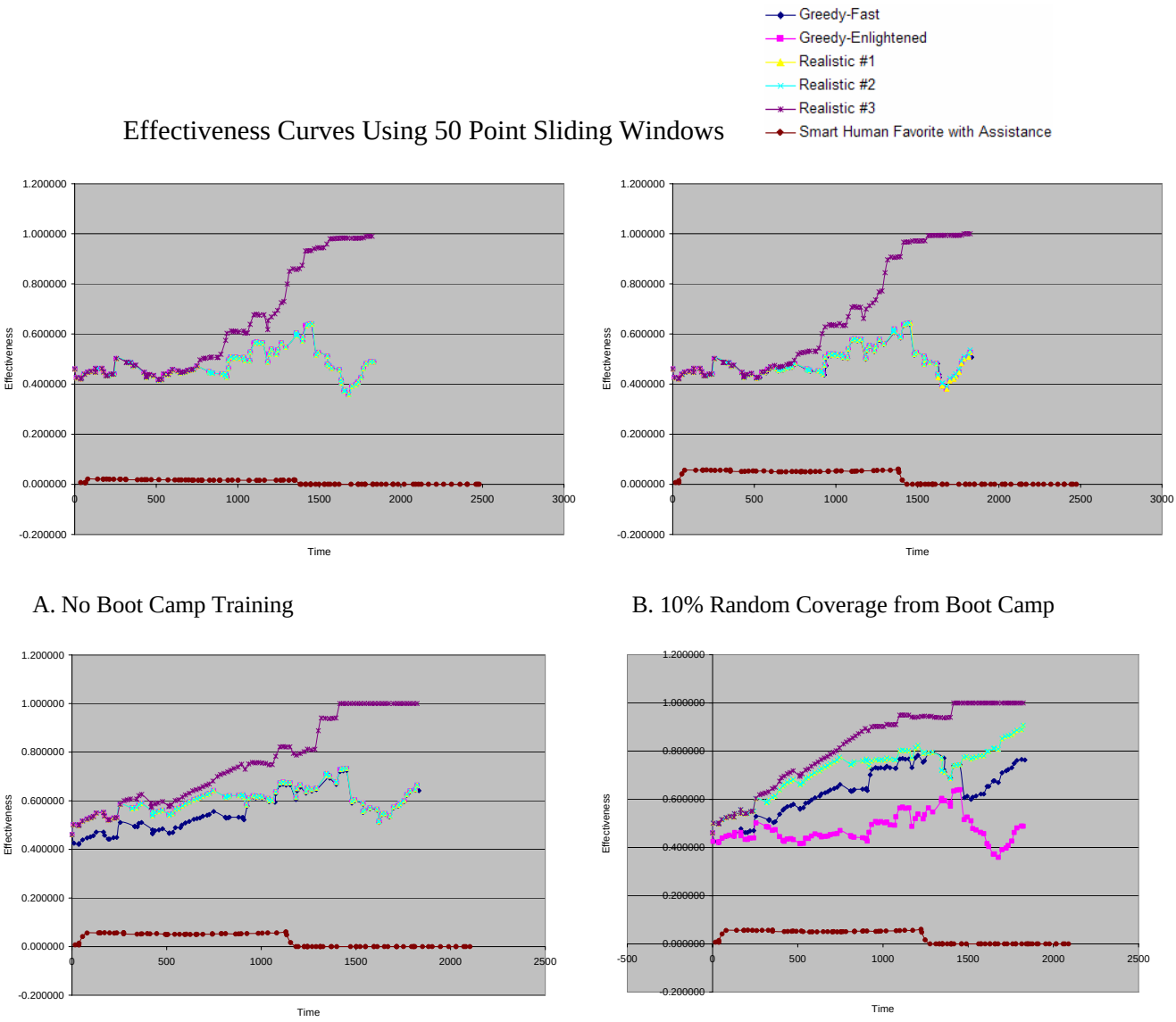

Figure 4 Effectiveness of Different Human Behavior Algorithms

CONCLUSIONS

The research presented here, demonstrates that a capability that must learn on-the-job, requires some prior knowledge before being deployed. The information discovered in this research can inform the design of a boot camp for assistants that learn. The results clearly demonstrate that more training is better, which is not surprising. What was less expected was the lack of contribution of fully known tasks to the results. The manipulated assistants that knew a small number of tasks completely rather than having broad incomplete knowledge were at a disadvantage in an environment that can include surprises and where the operations cannot be well predicted. Spare time graphs provide evidence that if one can determine some set of known tasks that the assistant can execute early in its deployment, thus saving time for the user, this can result in ultimate success provided the spare time generated is

used for further training. However, if the guess of which tasks are important to start with is wrong, the assistant is immediately put at a disadvantage to ones with broad random knowledge.

Some of the operations and conditions showed a considerable jump at the 20% level. Since knowledge in real world domain is for all intents and purposes infinite, what does it mean to have 20% coverage after the boot camp. In these experiments the knowledge that was available to be known was about the possible tasks for the operations. Therefore, providing an assistant with 20% of the knowledge needed associated with some list of tasks that are likely to occur should be sufficient to give the assistant a strong chance of not being abandoned.

Of less significance than expected was observability. When thinking about this factor, we can think of it in two ways. The experimental results show that learning by observation was a useful contributor to success. However, without other factors contributing, observability needs to be quite high before it by itself can prevent abandonment or disaster with the assistant. Explicit teaching was more beneficial, because it can be targeted to areas where it is more likely to help the cause, for instance the one remaining subtask that prevents an assistant from completely automating a method..

The environment was also looked at as a factor in success or failure of assistant deployment. The complexity of the domain (in terms of tree depth in the simulation) made no difference in the outcome. This means that the results found are likely valid for a broad range of environments.

Of considerable significance was the human behavior contribution. The results quite clearly show that learning capabilities require a different approach by the user. Success was only assured in all of the tests if the user followed an approach where spare time generated by the assistant was reinvested in the form of teaching the assistant about its environment and about tasks it might encounter in the future (represented by the Realistic #3 algorithm). Combined with sufficient knowledge from the boot camp to give a reasonable probability of early spare time, such a strategy provides significant confidence in success. Use of this strategy, significantly reduced variation, caused by random factors, in the likelihood of successful employment in our simulation, and could reasonably be assumed to do the same in an actual PAL deployment. This implies that in addition to finding ways of preparing the assistant, considerable attention must be paid to preparing the users before deployment.

The results of the human-favorite series of algorithms did show that a knowledgeable enough assistant might be able to prevent a user from failing despite a poor human work strategy. Analytically, one could see that if the human’s favorite way of doing things corresponded to the assistant’s knowledge, the human might not fall so far behind. This suggests a boot camp strategy that is very consistent with the PAL program vision. Users and their assistants must be together as early as possible, perhaps attending the boot camp together. If a well thought out plan for learning included the tasks most likely to be seen early in deployment was conducted so that the manner in which the user preferred to address the problem was the same as was taught to the assistant, one might have an optimal approach. This requires though, that the boot camp be able to set up conditions that allow the assistant to be taught in this way.

REFERENCES

Brachman, R., 2007. (AA)AI More than the Sum of Its Parts. AI Magazine 27(4): 19-34.

Chandrasekaran, B.; Johnson, T.; and Smith, J. 1992. Task-structure Analysis for Knowledge Modeling, Communications of the ACM 35(9): 124-139.

Cohen, P. and Pool., M. 2005. The CALO 2005 Experiment Data Analysis. http://calo.sri.com. Accessed 15 January 2007.

Lange, D.S., 2006a. PAL Boot Camp: Acquiring, Training, and Deploying Systems with Learning Technology. In Proceedings of the 2006 Command and Control Research and Technology Symposium, San Diego, Calif.: DOD CCRP.

Lange, D.S., 2006b. Boot Camp for Cognitive Systems. In Proceedings of the Twenty-First National Conference on Artificial Intelligence, 1889-1890. Boston, Mass.: AAAI Press.

Lange, D.S., 2007. Boot Camp for Cognitive Systems: A Model for Preparing Systems with Machine Learning for Deployment. Ph.D. diss.: Dept. of Computer Science, Naval Postgraduate School.

Meyers, K., 2006. Building an Intelligent Personal Assistant: The CALO Project. Proceedings of the Twenty-First National Conference on Artificial Intelligence, xliii. An Invited Talk. Boston, Mass.: AAAI Press. PowerPoint presentation found on 19 January 2007 at http://caloproject.sri.com/publications/downloads/AAAI06myers.ppt.

Murdock, J. 2000. Semi-Formal Functional Software Modeling with TMK, Technical Report GIT-CC-00-05, College of Computing, Georgia Institute of Technology.

Wong, L.; Lange, D.S.; Sebastyn, J.T.; and Roof, W.H. 2006. Command World. In Proceedings of the 2006 Command and Control Research and Technology Symposium, San Diego, Calif.: DOD CCRP.