AI Magazine Volume 26 Number 3 (2005) (© AAAI)

Identifying Terrorist Activity with AI Plan-Recognition Technology

Peter A. Jarvis, Teresa F. Lunt, and Karen L. Myers

■ We describe the application of plan-recognition techniques to support human intelligence analysts in processing national security alerts. Our approach is designed to take the noisy results of traditional data-mining tools and exploit causal knowledge about attacks to relate activities and uncover the intent underlying them. Identifying intent enables us to both prioritize and explain alert sets to analysts in a readily digestible format. Our empirical evaluation demonstrates that the approach can handle alert sets of as many as 20 elements and can readily distinguish between false and true alarms. We discuss the important opportunities for future work that will increase the cardinality of the alert sets to the level demanded by a deployable application. In particular, we outline the need to bring the analysts into the process and for heuristic improvements to the plan-recognition algorithm.

Events in the United States during 2001 tragically demonstrated the nation's vulnerability to acts of terrorism. U.S. security agencies had information available at that time that could have been used to thwart the World Trade Center, Pentagon, and Shanksville, Pennsylvania, attacks. However, that information was not utilized because it was in an ocean of total intelligence leads then under consideration (U.S. Senate Report 2002).

Significant research has focused on the problem of uncovering the critical pieces of intelligence information that can be used to thwart an attack from a large body of intelligence leads (DISCEX 2003). The data-mining approaches that have been explored for this purpose can sift through vast quantities of information but suffer from a high false alarm rate and do not help analysts link together separate facts and events (Ning and Dingbang 2003). We have developed a proof-of-concept prototype for a tool to automate the analysis currently undertaken by humans by exploiting plan-recognition techniques from the automated planning community. Our thesis is that we can significantly improve the quality of the information passed to human analysts if we can automatically discover a significant causal coherence among disparate activities. Our analysis can also aid in the explanations of hypotheses by presenting them in the context of the evidence.

We structure this article as follows. We first present the computer-aided plan-recognition (CAPRe) architecture. We then describe the modeling framework that we use to represent terrorist behavior before detailing the plan-recognition algorithm we have developed to match observations with the model. Our experimental section presents an evaluation of the performance of the system on alert sets with a range of signal-to-noise properties. In particular, the results show that while our approach has great promise it does not currently scale to

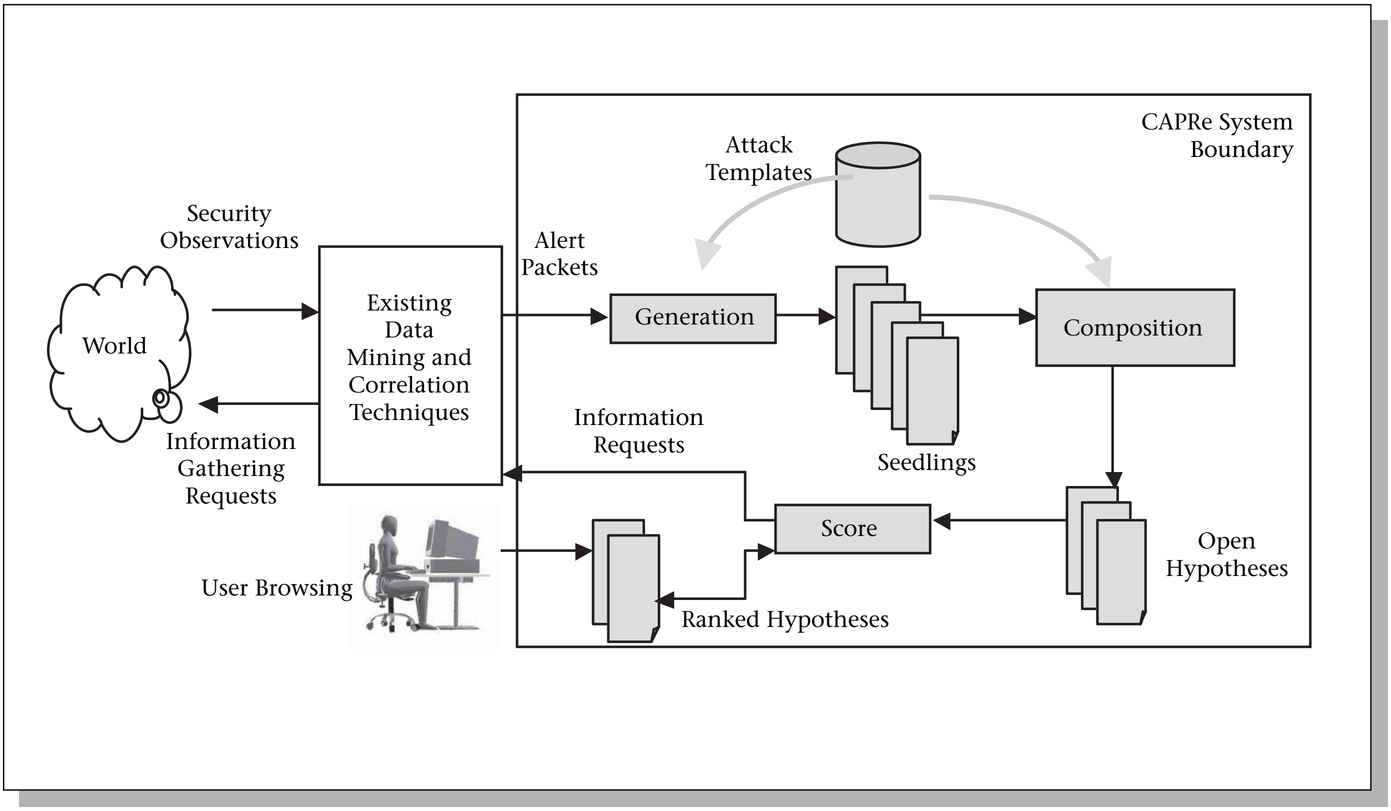


Figure 1. CAPRe Architecture.

the level demanded by a deployed application. We close by reflecting on what we have learned and define avenues that must be explored by future work before the capability can be deployed in an operational setting.

Computer-Aided Plan-Recognition (CAPRe) Architecture

Our approach, illustrated in figure 1, involves specifying a set of a priori attack templates that describe possible attack strategies in terms of observable actions and effects. CAPRe is fed alert packets made up of security observations that have been clustered together by standard correlation and data-mining techniques. CAPRe applies a two-phased plan-recognition process to this observation set using the attack templates to generate a (possibly empty) set of hypothesized attack goals that may be under way. This process involves first identifying individual observations that match actions or effects in the attack templates (seedling generation) and then aggregating these observations into larger, coherent sets (composition). The results are then ranked and presented to the user, along with information requests that recommend targeted investigations to confirm or refute the hypothesized attack goals. Note that the templates support the anticipation of future steps, thus enabling early interventions to block in-progress attacks.

Attack Templates

CAPRe's attack template library contains a description of attack activities structured hierarchically with a specification of the conditions under which they can be combined. The library forms a description of the action physics for a particular application domain that can be used to construct plan instances tailored to specific target requirements. It is not a library of known attack plan cases.

We draw on the rich hierarchical action representation developed and refined in the automated planning community during the past 30 years (Fikes and Nilsson 1971, Tate 1977). These representations have found application in areas as diverse as spacecraft control (Muscettola et al. 1997) and oil-spill response planning (Bienkowski, des Jardins, and Desimone 1994). Figure 2 presents a sampling of the templates in our library for terrorist attacks on a national infrastructure. A template contains information organized into six slots: vars,

```
:template Physical_Attack
   :vars group ?group, target ?target;
   :purpose destroy(?group,  ?target)
   :tasks
      1. reconnaissance(?group, ?target)
      2. prepare_attack(?group, ?target)
      3. attack(?group, ?target);
   :orderings1 --> 3, 2 --> 3;
   :end_template

   :template Reconnaissance_of_Target
   :vars group ?group, target ?target;
   :purpose reconnaissance(?group, ?target)
   :tasks
      1.   recon_security(?group, ?target)
      2.   recon_structure(?group, ?target);
    :end_template

   :template Research_Structure
   :vars group ?group, target ?target,
         person ?person1, ?person2, ?person3,
         engineering_school ?engineering_school;
   :purpose recon_structure(?group, ?target)
   :tasks
      1. obtain_control_information (?group, ?target)
      2. take_job (?group, ?target)
      3. structural_engineering_training (?group, ?engineering_school);
   :conditions member(?person1, ?group), member(?person2, ?group), member(?person3, ?group);
   :effects  retrieved_blueprints(?person1, ?target) :at 1
                frequency: low
                accuracy: high
                gathering_cost: high.
             hr_records(?person2, ?target) :at 2
                frequency: high
                accuracy: high
                gathering_cost: low.
             enrollment(?person3, ?engineering_school) :at 3
                frequency: high
                accuracy: high
                gathering_cost: low.
    :end_template
```

Figure 2. Example Templates.

purpose, tasks, orderings, preconditions, and effects.

The :vars slot consists of variables that provide typed parameter descriptions for a template. The :purpose slot defines the overall purpose of the template. The :tasks slot consists of a set of labeled lower-level tasks to be performed to achieve the template's purpose. The :orderings slot contains temporal constraints on the execution of actions, defined in terms of task labels. The :preconditions slot consists of constraints on the execution of a task (such as the requirement for a valid driver's license to rent a car). Finally, the :effects slot consists of changes to the world that result from the execution of tasks within a template.

We define three properties on each task and effect within a template, *frequency, accuracy,* and *gathering cost.* Each property can take the value high, medium, or low. While more complicated schemes are possible, we decided that this simple scheme would be the most accessible to our user community.

The first property consists of the frequency

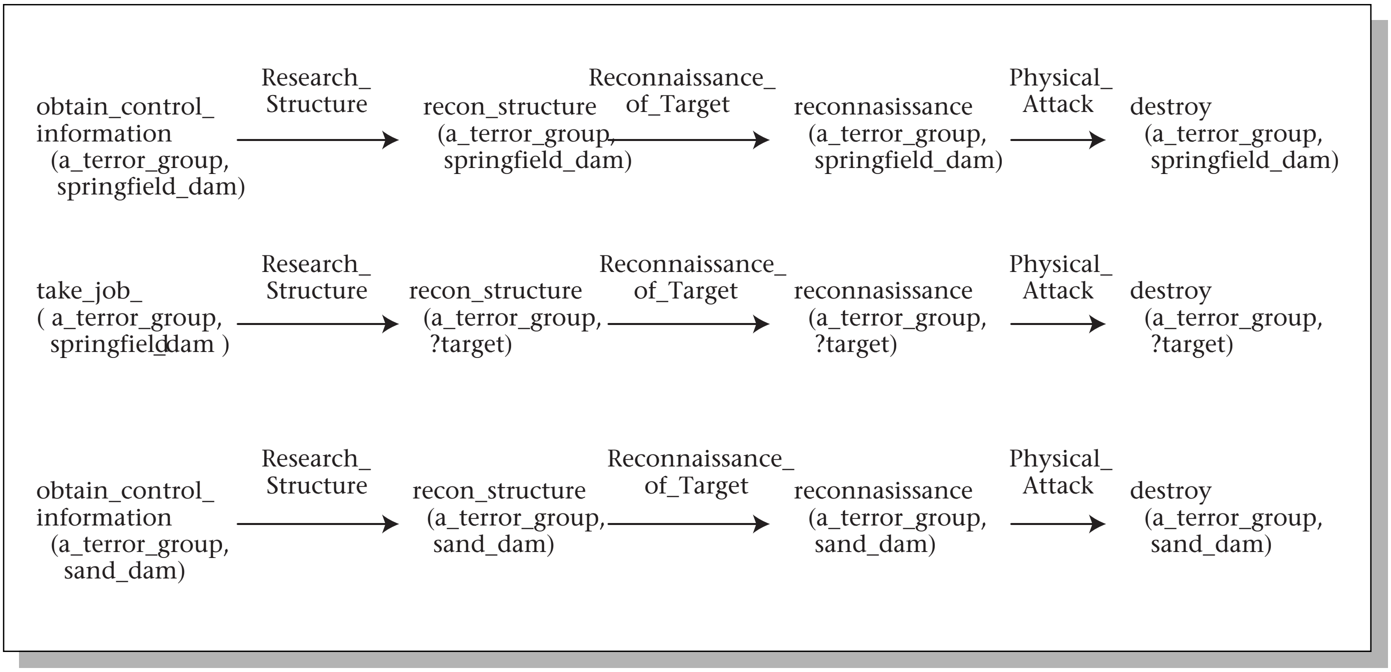

obtain_control_ information (a_terror_group, springfield_dam)
Research_ Structure
recon_structure (a_terror_group, springfield_dam)
Reconnaissance_ of_Target
reconnasissance (a_terror_group, springfield_dam)
Physical_ Attack
destroy (a_terror_group, springfield_dam)
take_job_ (a_terror_group, springfield_dam)
Research_ Structure
recon_structure (a_terror_group, ?target)
Reconnaissance_ of_Target
reconnasissance (a_terror_group, ?target)
Physical_ Attack
destroy (a_terror_group, ?target)
obtain_control_ information (a_terror_group, sand_dam)
Research_ Structure
recon_structure (a_terror_group, sand_dam)
Reconnaissance_ of_Target
reconnasissance (a_terror_group, sand_dam)
Physical_ Attack
destroy (a_terror_group, sand_dam)

Figure 3. Example Seedlings.

of the occurrence of a task or effect in normal behavior. For example, car rentals are assigned a high frequency, while missing person reports are assigned a low frequency.

The second property is accuracy of normal observations of a task or effect. For example, missing person reports are highly accurate, while a witness's recollection of a suspicious car's license tag is generally of low accuracy.

The final property, gathering cost, records the cost of making an observation. Accessing an online database is considered low cost, while an observation that demands a door-to-door search by law enforcement officials is high cost.

Frequency and accuracy properties are exploited during the plan-recognition process to score hypotheses or to filter observation lists. Gathering cost is used during the information-gathering planning phase to determine the cost benefit of a particular information-gathering action.

Plan-Recognition Process

We first provide an overview of our plan-recognition process before describing each element in detail. The broad approach is adapted from Karen Myer's earlier work on abductive plan sketch completion (Myers 1997).

Process Overview

Informally, the plan-recognition task is to take a set of observed actions and world state changes and a collection of (attack) templates and produce a set of plans that offer potential explanations for the observations within the template set.

Consider the observation *retrieved_blueprints (john_doe, springfield_dam)* in the context of the templates shown in figure 2. We can match this observation to an effect and its associated task in the Research_Structure template to conclude that the group of which John Doe is a member is attempting to obtain control information about the Springfield dam. Working back two steps, we can explain the control information attempt as part of the broader reconnaissance component of a physical attack on this dam. Figure 1 shows this stage as the generation phase of the plan-recognition activity that results in a set of seedling explanations for each observation.

The composition phase of the plan-recognition activity takes the set of seedlings generated for all observations and seeks subsets that can be combined consistently with respect to the templates. For example, if we had observed another individual with links to the same organization as John Doe taking a job at the dam, then we could combine these two observations to form an open hypothesis. If John Doe and this new individual were associated with two different organizations, then these seedlings would not be able to be combined as they would violate the member conditions in the

Research_Structure template. However, in this case each seedling would still stand as an explanation in its own right.

We now describe these two phases in more detail.

Seedling Generation

We use an abductive inference procedure to identify the seedling hypotheses that provide candidate explanations for a set of observed activities. Each seedling hypothesis tops out in an element of the goal space, *G,* for a domain model. While *G* could be defined explicitly, we use the set of templates with purpose fields that do not appear as tasks in other templates. *Destroy(?group, ?target)* is the sole member of *G* given our example template set and is shown in figure 2.

Definition 1 (Task Seedling)

The seedling set for a task *A* is the set of labeled linear graphs

$A\ O_n,\sigma_n\ G_n\ O_{n-1},\ \sigma_{n-1}\ G_{n-1}\ O_{n-2},\sigma_{n-2} \ldots O_1,\sigma_1\ G_1$

where $G_1 \in G$, and O_j is a template with purpose G_j and a subtask T such that σ_j is a most general unifier of T and $(G_{j+1})^\beta$, for $\beta = \cup_{n \leq i \leq j} \sigma_j$ and $A = G_{n+1}$.

A corresponding notion of effect seedling can be defined, where the root of the linear graph is an effect rather than a task for the template O_n. For example, the observation *retrieved_blueprints (john_doe, springfield_dam)* is an effect seedling for the *obtain_control_information(a_terror_group, springfield_dam* action in the Research_Structure template. This assumes that we know that John Doe is a member of and only a member of the group known as the "A Terror Group." CAPRe can handle both action and action effects appearing in security alerts.

Consider the following observation set (the effect preprocessor has been applied):

{*obtain_control_information(a_terror_group, springfield_dam),*

take_job(a_terror_group, springfield_dam),

obtain_control_information(a_terror_group, sand_dam)}.

Figure 3 shows the seedlings that are generated for these observations given the template set shown in figure 2. Technically, this figure should show variable bindings, but we have simplified the presentation to the propositional case.

1. {obtain_control_information(a_terror_group, springfield_dam), take_job(a_terror_group, springfield_dam) }
2. {obtain_control_information(a_terror_group, springfield_dam), obtain_control_information(a_terror_group, sand_dam)}
3. {obtain_control_information(a_terror_group, sand_dam), take_job(a_terror_group, springfield_dam)}
4. {obtain_control_information(a_terror_group, springfield_dam), take_job(a_terror_group , springfield_dam), obtain_control_information(a_terror_group, sand_dam)}

Figure 4. The Sets in the Powerset of SG.

Implementing a seedling-generation procedure is simple given the above definition. The only concern is the run time of the procedure. We define the abstraction factor for a task *T* to be the number of template schema subtasks that unify with *T.* We define α to be the maximum abstraction factor for all the tasks in a domain definition. Let *la* be the difference in abstraction level between observations *a* and the most abstract goal in the goal space *G*. The sum

$$\sum_{a \in \text{observation}} \alpha^{la}$$

is a loose upper bound on the construction time for the seedling explanations for a set of observations.

We show empirically the time spent in the seedling-generation phase in our experiment section. While a procedure with exponential bounds is cause for concern, we have found in practice that typical domain definitions contain 8 to 10 abstraction levels, and the abstraction factor rarely exceeds 6 (Myers 1997).

We provide two mechanisms that allow the user to influence the seedling-generation process. The frequency filter parameter allows the user to specify the maximum occurrence frequency of the observations that should be considered. This allows the user to filter out high- or medium-frequency events. The user can also specify classes of events and effects to ignore.

Seedling Composition

The composition phase seeks to combine the seedlings generated in the first phase to form a set of open hypotheses with each member offering an explanation of the intent behind a set of seedlings. Considering figure 3, all three seedlings would be clustered into a single seedling set, *SG,* as they share the common destroy top-level predicate. We now iterate through the powerset of each seedling set (ignoring those of cardinality < 2 and generating the set incrementally) to identify the seedlings that can be combined. The powerset of *SG* that

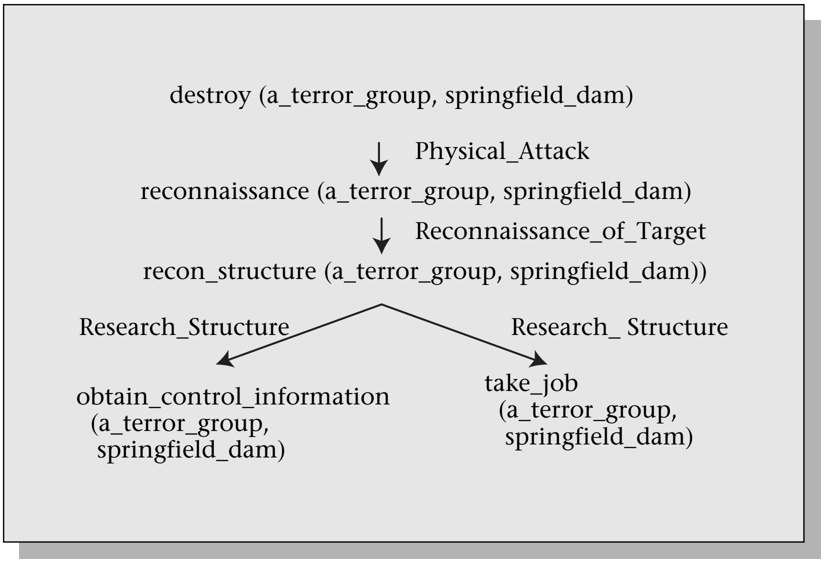

destroy (a_terror_group, springfield_dam)
Physical_Attack
reconnaissance (a_terror_group, springfield_dam)
Reconnaissance_of_Target
recon_structure (a_terror_group, springfield_dam))
Research_Structure
Research_ Structure
obtain_control_information (a_terror_group, springfield_dam)
take_job (a_terror_group, springfield_dam)

Figure 5. Example Open Hypothesis.

we consider consists of the sets shown in figure 4 (we show only the observation that starts each seedling).

Corresponding seedling steps can be combined to form an open hypothesis if the following conditions are satisfied: (1) the purpose statements unify; (2) the steps use a common template; (3) all constraints in the template are satisfied; and (4) the bindings entailed by the purpose and task statements across the steps are consistent.

The combination process starts with the top-level step of each seedling in the set under consideration. Consider the members of set 1. The top-level steps can be combined as the conditions from both seedlings are satisfied. Now consider the members of set 2. The top steps of these two seedlings cannot be combined, as the binding for the target variable is inconsistent across the seedlings. Set 3 cannot be combined as bindings for the target variable at the top of the seedlings are different. Return to set 1; the combination process continues by considering the next steps in each seedling under consideration. A valid open hypothesis is produced if all steps in all seedlings could be combined. Figure 5 shows the open hypothesis produced by combining the members of set 1.

The concern with the composition procedure, as with the seedling-construction phase, is the computational complexity of the procedure given that the size of a power set of *n* elements is 2^n.

We carefully structure our search and exploit search pruning to maximize the number of seedlings that we can consider. Our primary strategy is to search through the powerset of seedlings in ascending cardinality order. We first search through all subsets of cardinality 1, then 2, and then 3, and so on. This strategy has several advantages. First, we can terminate our search as soon as we find a cardinality level with no open hypotheses. Second, we can use a nogood recording strategy (Dago and Verfaillie 1996). Once we have found a set of seedlings that cannot be combined, we can prune all other sets that contain that set as a subset.

Implementation

We have implemented a prototype of our CAPRe architecture in Java. Figure 6 shows the user interface to this system. The top pane displays the current open hypotheses with observations bolded. The left and center bottom panels allow the user to control the seedling-generation and composition processes and the scoring function used to sort the open hypotheses. We currently support a simple scoring scheme that rates open hypotheses according to the number of observations that support them. The bottom right panel displays the constraints on the currently selected hypotheses.

Our current implementation includes only the plan-recognition portion of the architecture. Implementing and evaluating the generation of information-gathering plans is left for further work.

Experiments

Our empirical evaluation of the CAPRe implementation examined the performance of the system on a range of alert sets. We focused on variations in the following properties of alert sets: (1) Number of alerts is the total number of alerts in a set. (2) Signal-to-noise ratio is the number of alerts in a set that are part of a malicious plan (the target plan) divided by the total number of false alerts in the set. A false alert is an observed action that is not part of an attack. (3) Noise coherence is the maximum number of false alerts in a set that can be combined consistently to form a coherent attack plan.

Table 1 presents the results of our empirical examination of CAPRe. The input alert sets were crafted by hand to include evidence for a target hypothesis together with noise with the appropriate signal-to-noise and coherence properties. We recorded both the run time of the system and the position of the target hypothesis within the sorted list of hypotheses identified for each set (called the *rank*). The rank of a hypothesis is a simple function of the number of seedlings that have been combined

Noise Coherence		1			4			8		
No. Events	Signal/ Noise	1/3	1/1	3/1	1/3	1/1	3/1	1/3	1/1	3/1
8	**Time**	0:00:00.1	0:00:00.1	0:00:00.1	0:00:00.1	0:00:00.1	0:00:00.1	0:00:00.1	0:00:00.1	0:00:00.1
	Rank	**1/7**	**1/5**	**1/3**	3/3	**1/2**	**1/2**	2/2	*Joint 1st*	**1/2**
16	**Time**	0:00:00.7	0:01:00.4	0:01:26.0	*0:00:01.7*	0:01:02	0:01:25.9	0:03:06.1	0:01:00.1	0:00:23.8
	Rank	**1/13**	**1/9**	**1/4**	*Joint 1st*	**1/3**	**1/2**	3/3	*Joint 1st*	**1/2**
20	**Time**	0:00:09.7	2:27:02.4	0:47:34.0	0:00:06.3	0:47:28.2	0:48:19.4	0:28:01.0	0:48:20.5	0:54:59.4
	Rank	**1/16**	**1/12**	**1/8**	5/5	**1/4**	**1/3**	3/3	**1/2**	**1/2**
25	**Time**	0:19:49.3	*	*	*	*	*	*	*	*
	Rank	**1/19**	*	*	*	*	*	*	*	*

Table 1. Experimental Results

Apple PowerMac G5 1.8 GHz, 500 MB RAM. *Denotes no result after 12 hours.

to support it. This is intuitive, as the more evidence available for a hypothesis the higher its rank.

We sorted the set of hypotheses by the number of seedlings combined to generate hypotheses. The more seedlings composed to form a hypothesis, the higher its support, and therefore the higher its score.

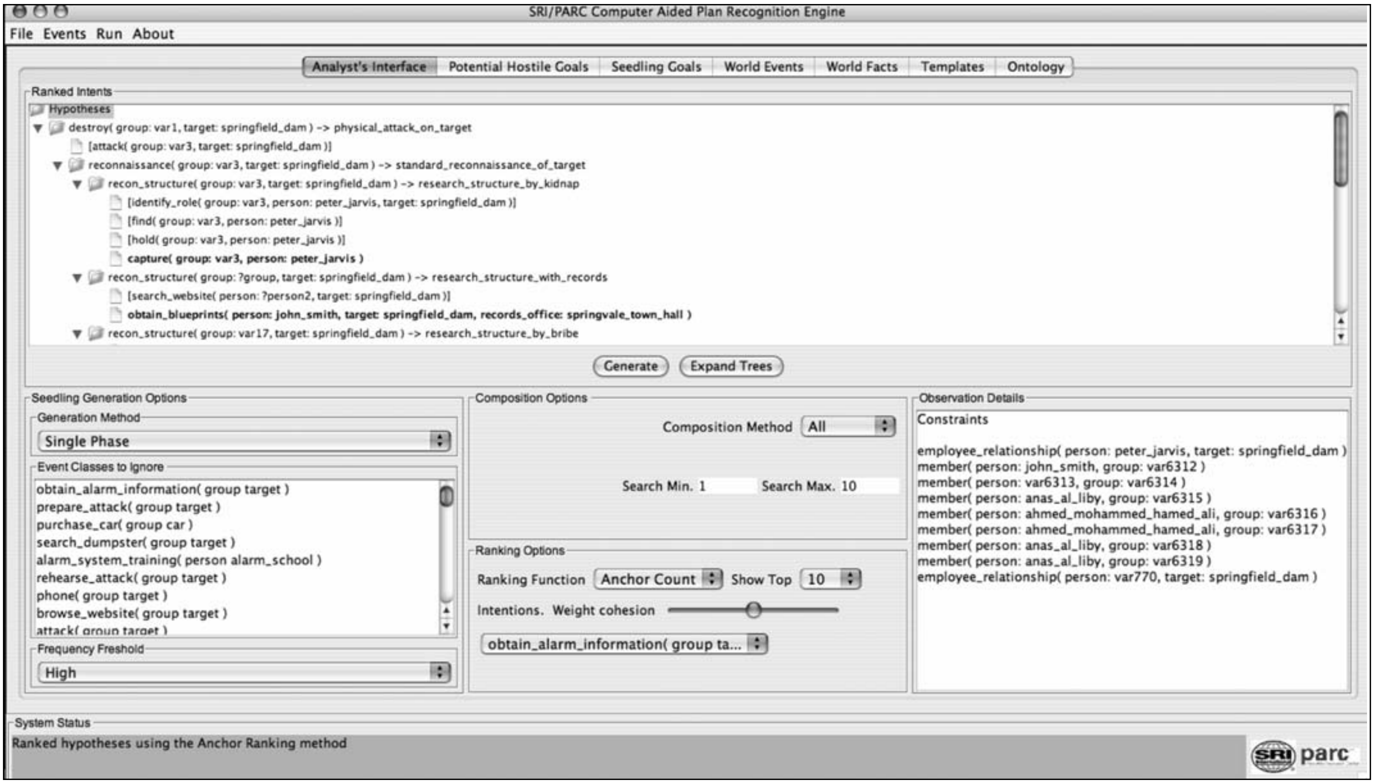
Examining table 1 reveals that the time CAPRe takes to identify intent increases exponentially with the number of events in an alert set. This is the behavior that we predicted given that our seedling-composition step must consider the powerset of the seedlings generated for an alert set. CAPRe is currently limited to alert sets of about 20 actions on state-of-the-art hardware.

The noise coherence and signal-to-noise ratio properties of alert sets affected both the run time and accuracy of CAPRe. Consider first the group of three results with a noise coherence of 1. In this case, noise cannot mislead the recognition process as each mistaken hypothesis can be supported by only one alert. In this situation, CAPRe consistently ranks the target hypothesis first. When we move to alerts with a noise coherence of 4, CAPRe ranks incorrect hypotheses higher when the signal-to-noise ratio favors the noise. The target hypothesis is ranked first again when the signal-to-noise ratio is 1 or favors the signal. When the noise coherence is adjusted to 8, the results show that it becomes increasingly difficult for CAPRe to correctly identify the target hypothesis. This is understandable, as the noise has become coherent and is dwarfed in cases where it outnumbers or equals the target activities.

We draw two conclusions from the experiments. First, the run-time performance degrades exponentially with alert set size, and for practical purposes 20 alerts is the limit. Second, the accuracy of the system falls off as the cohesion of the noise exceeds that of the actual attack activity.

Summary and Further Work

We have introduced the CAPRe architecture for automating the deep analysis of security alert clusters in order to reduce the load on human security analysts. Our proof-of-concept demonstration illustrates that the technology is capable of recognizing the intent behind events in an alert set and of presenting that intent succinctly to a human user. Our empirical investigation concluded that the technology could process alert clusters of as many as 20 actions and demands that noise (false alerts) be less causally coherent than the components of the attacks.



Ranked Intents

```
Hypotheses
▼ destroy( group: var1, target: springfield_dam ) -> physical_attack_on_target
    [attack( group: var3, target: springfield_dam )]
  ▼ reconnaissance( group: var3, target: springfield_dam ) -> standard_reconnaissance_of_target
    ▼ recon_structure( group: var3, target: springfield_dam ) -> research_structure_by_kidnap
        [identify_role( group: var3, person: peter_jarvis, target: springfield_dam )]
        [find( group: var3, person: peter_jarvis )]
        [hold( group: var3, person: peter_jarvis )]
        capture( group: var3, person: peter_jarvis )
    ▼ recon_structure( group: ?group, target: springfield_dam ) -> research_structure_with_records
        [search_website( person: ?person2, target: springfield_dam )]
        obtain_blueprints( person: john_smith, target: springfield_dam, records_office: springvale_town_hall )
    ▼ recon_structure( group: var17, target: springfield_dam ) -> research_structure_by_bribe
```

Seedling Generation Options

Generation Method: Single Phase

Event Classes to Ignore

```
obtain_alarm_information( group target )
prepare_attack( group target )
purchase_car( group car )
search_dumpster( group target )
alarm_system_training( person alarm_school )
rehearse_attack( group target )
phone( group target )
browse_website( group target )
attack( group target )
```

Frequency Freshold: High

Composition Options

Composition Method: All

Search Min. 1 Search Max. 10

Ranking Options

Ranking Function: Anchor Count Show Top: 10

Intentions. Weight cohesion

obtain_alarm_information(group ta...

Observation Details

```
Constraints

employee_relationship( person: peter_jarvis, target: springfield_dam )
member( person: john_smith, group: var6312 )
member( person: var6313, group: var6314 )
member( person: anas_al_liby, group: var6315 )
member( person: ahmed_mohammed_hamed_ali, group: var6316 )
member( person: ahmed_mohammed_hamed_ali, group: var6317 )
member( person: anas_al_liby, group: var6318 )
member( person: anas_al_liby, group: var6319 )
employee_relationship( person: var770, target: springfield_dam )
```

System Status

Ranked hypotheses using the Anchor Ranking method

Figure 6. CAPRe User Interface.

Further work is needed to move CAPRe to a position where it is ready for operational deployment. We recommend that further work focus on the following three areas.

The first area of focus is real alert sets. CAPRe has benefited from a close development relationship with the intelligence community. However, it is essential that future work have access to actual or at least analyst-generated alert sets. Research can then focus on addressing the issues raised by actual rather than projected signal-to-noise ratios and coherence factors.

The second area of focus is the mixed-initiative paradigm. The number of seedlings generated for an alert set is the critical factor in determining the run time of the composition phase. Intelligence analysts often have deep insights into the attack activity in progress and events that are supporting false conclusions. Future work should explore a mixed-initiative approach where seedlings are presented in a digestible form to the analysts for filtering.

A final area of focus is heuristic development. We propose to explore two approaches to improving the algorithmic performance of CAPRe. First, we will explore the information-gathering planning concept shown in the architecture diagram to enable us to perform plan recognition on low-frequency actions in an alert cluster before examining the cluster for alerts that support the set of hypotheses generated. This approach would have the key benefit of reducing the number of seedlings generated for an alert set. Second, we will explore the inclusion of probabilities of observing template tasks given a template purpose in a way similar to that used by Goldman, Geib, and Miller (1999). We will use this information to rank seedlings according to the probability that the observation supports the goal of each. A simple cutoff strategy can then be used to prune unlikely seedlings and again reduce the number of seedlings passed into the computationally expensive combination phase.

Acknowledgments

We thank Tom Garvey and other members of SRI's AI Center who have commented on this work. The DARPA Cyber Panel program through subcontract MP201427 under contract N66001-03-C-8004 supported the work. Much of the work reported in this article was undertaken while Peter Jarvis was at SRI International.

References

Bienkowski, M.; des Jardins, M.; and Desimone, R. 1994. SOCAP: System for Operations Crisis Action Planning. Paper presented at the ARPA/Rome Lab 1994 Knowledge-Based Planning and Scheduling Initiative Workshop, February.

Dago, P., and Verfaillie, G. 1996. Nogood Recording for Valued Constraints Satisfaction Problems. In *Proceedings of the Eighth International Conference on Tools with Artificial Intelligence (*ICTAI'96). Los Alamitos, CA: IEEE Computer Society.

DISCEX. 2003. *Proceedings of the Third DARPA Information Survivability Conference and Exposition* (DISCEX-III 2003), ed. D. Maughan. Los Alamitos, CA: IEEE Computer Society.

Fikes, R., and Nilsson, N. 1971. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence* 2(3–4): 189–208.

Goldman, R.; Geib, C.; and Miller, C. 1999. A New Model of Plan Recognition. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, ed. C. Meek and U. Kjaerulff, 245–254. San Francisco: Morgan Kaufmann Publishers.

Muscettola, N.; Smith, B.; Fry, C.; Chien, S.; Rajan, K.; Rabideau, G.; and Yan, D. 1997. On Board Planning for Autonomous Space Craft. In *Proceedings of the IEEE 1997 National Aerospace Conference* (NAECON 1997). Piscataway, NJ: Institute of Electrical and Electronic Engineers.

Myers, K. 1997. Abductive Completion of Plan Sketches. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence* (AAAI-97). Menlo Park, CA: AAAI Press.

Ning, P., and Dingbang, X. 2003. Adapting Query Optimization Techniques for Efficient Intrusion Alert Correlation. In *Proceedings of the Seventeenth IFIP WG 11.3 Working Conference on Data and Application Security*. Dordrecht, The Netherlands: Kluwer Academic Publishers.

Tate, A. 1977. Generating Project Networks. In *Proceedings of the Fifth International Joint Conferences on Artificial Intelligence*, 888–893. Los Altos, CA: William Kaufmann, Inc.

U.S. Senate Select Committee on Intelligence and U.S. House Permanent Select Committee on Intelligence. 2002. *Joint Inquiry Into Intelligence Community Activities Before and After the Terrorist Attacks of September 11, 2001.* Senate Report No. 107-351, December. Washington, DC: Government Printing Office. (www.gpoaccess.gov/serialset/creports/911.html). Accessed October 27, 2004.

Peter Jarvis (pjarvis@mail.arc.nasa.gov) has been working on the application of automated planning technologies for 10 years. He spent 3 years on the O-Plan team at the University of Edinburgh where he developed a U.S. Army small unit planning application. He then moved to SRI International where he spent 5 years working on U.S. Air Force and Special Forces planning applications. Jarvis is now at NASA Ames Research Center where he is using plan-generation and repair techniques to automate the testing of spacecraft crew operating procedures. He holds a Ph.D. in artificial intelligence planning from the University of Brighton, UK.

Teresa F. Lunt, an expert in information security and information warfare, manages the Computer Science Laboratory at the Palo Alto Research Center. Before joining PARC, she was associate director of the Computer Science Laboratory at SRI International and an assistant director and program manager at DARPA. At DARPA, Lunt developed and managed the Information Survivability program, was instrumental in developing the Information Assurance program, and served as DARPA's point of contact for coordination with the National Security Agency and other DARPA programs. She is a member of IEEE, the IEEE Computer Society, the Association for Computing Machinery, the International Federation for Information Working Group 11.3 on database security and Working Group 10.4 on reliability, and the IEEE Computer Society Technical Committee on Security and Privacy. She has served on the Air Force Scientific Advisory Board and is the recipient of a number of prestigious awards. She received her M.A. degree in applied mathematics from Indiana University.

Karen Myers is director of the Intelligent Mixed-initiative Planning and Control Technologies (IMPACT) program within the AI Center at SRI International. Myers joined SRI in 1991 after completing a Ph.D. in computer science at Stanford University. Her research interests include the areas of reactive control, multiagent systems, automated planning, advisable technologies, and mixed-initiative problem solving. Her work in these areas spans the range of basic research,