Creative Problem Solving Through Automated Planning and Analogy
Paper type: Technical Paper

Richard G. Freedman, Scott E. Friedman, David J. Musliner, and Michael J.S. Pelican
SIFT, LLC
Minneapolis, MN USA
{rfreedman,sfriedman,musliner,mpelican}@sift.net

Abstract

As a creative problem solving strategy, analogical reasoning helps generalize and transfer solutions to new domains. Automated planners have been used for problem solving, but they reach *impasses* when their representation of the problem space lacks operators or resources to generate a plan from an initial condition to a set of goal conditions. This paper presents Creative Problem Solver (CPS), a novel integration of automated planning and analogical reasoning that recognizes when analogical reasoning and generalization may resolve such impasses. CPS uses heuristics to identify missing critical resources and then uses analogical reasoning to identify suitable replacements that are readily available in the environment. We implement CPS using the CIRCA planning architecture, which supports domain-level and meta-level planning, and the structure-mapping model of analogical reasoning. This paper demonstrates CPS resolving impasses in domains where traditional automated planning would otherwise fail.

1 Introduction

Planners often fail to find solutions because they lack operations or resources required to solve a problem. One way to overcome these challenges is for the planner to *generalize* from an ideal tool, resource, or prototype to a similar, available tool or resource. Generalizing via *analogy*, which facilitates comparative analysis via structure-mapping (Gentner 1983; McLure, Friedman, and Forbus 2015), requires no rules or operators, so we believe this is a practical, tractable method for generalization during planning. We have developed the Creative Problem Solver (CPS) system by integrating analogical structure-mapping with the CIRCA classical planning framework (Musliner, Durfee, and Shin 1993). This paper presents our technical approach and early empirical results with this planning-with-analogy strategy.

CPS integrates classical planning with analogy-based generalization in the following fashion:

1. CPS's domain-level planner encounters an impasse during planning and reports it to the higher-level *configuration planner* (CP).
2. CPS has a description of an ideal (but unavailable) resource that would resolve the impasse if it were available. The description includes details of the *affordances* of the resource.
3. CPS's analogical reasoner matches the ideal resource description against available resources, computing isomorphic structure (*i.e.*, matching relations and parameters), non-isomorphic structure, and similarity scores.
4. CPS attempts to use the results of analogical reasoning to resolve its impasse by generating new domain operator(s) that utilize the available resource(s).

These operations ultimately devise solutions to planning problems when the planner would otherwise be blocked by a missing operator or resource. For example, consider planning to protect a solar collector from a potential sandstorm on Mars. Suppose that traditional planning encounters an impasse for lack of a **solar collector cover** to use with a **cover** operator. In this case, analogical reasoning could map the ideal **solar collector cover** and its affordances against available resources to identify a sheet of habitat flooring or a quilt of spacesuit fabric as possible solutions with isomorphic structure.

We briefly review some relevant technical challenges involved in this creative problem-solving behavior, as well as factors that address these in our CPS approach.

Recognizing opportunities for generalization: When planners reach an impasse (a deadend where no current subgoal can be achieved by any operator), they generally backtrack to change a prior decision, potentially using dependency reasoning to determine the most recent decision that affects the impasse (Goldman, Pelican, and Musliner 2004). As CPS performs conventional planning, it stores *potential analogical impasse points* (*e.g.*, due to missing resources) and will only investigate them if conventional planning fails. Missing resource impasses are identified by preconditions of actions that *would* achieve a subgoal, if the resource was available. Storing these impasses for later investigation allows the standard planning search to continue, only invoking more complex and uncertain analogical reasoning if the planning fails entirely.

Recognizing resources that generalize: Given a missing resource in a plan, how do we assess the suitability of various available substitutes? In the absence of more intricate background knowledge and common sense, CPS uses *semantic similarity* to approximate generalizability. CPS uses a structure-mapping model of analogical reasoning to compute isomorphisms over semantic relations, attributes, and numerical parameters to rank available resources' ability to generalize against the missing, desired resource. We demonstrate in our results that structure-mapping is a suitable

heuristic for approximating generalizability, and we also outline possible improvements as future work.

Resolving an impasse with a generalization: Identifying generalizable resources does not, in itself, solve an impasse; CPS must revise its planner's configuration to accommodate the generalization. CPS generates a new operator that uses the proposed resource instead of the ideal resource. This leads to a new planning domain model, and a re-start of the domain-level planner.

Following a brief background on CIRCA, in Section 2 we introduce a domain that benefits from a planner that exhibits creative problem solving. Section 3 describes extensions to CIRCA to identify missing resources and find alternatives via analogy. We then illustrate how CPS addresses related problems from the example domain in Section 4, and we evaluate CPS's present scalability across multiple dimensions in Section 5. We describe related work in Section 6, and directions for future work in Section 7.

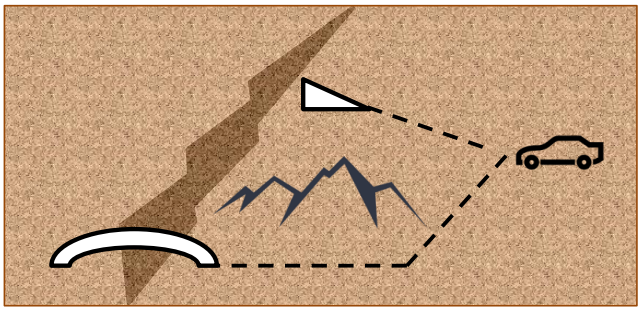

Figure 1: In the Dukes of Hazzard domain, an autonomous car must travel across a river canyon from East to West.

2 Background

As a proof of concept for our approach to plan generalization through analogical reasoning about resources in classical planning domains, we have adapted our CIRCA planning system to use analogical reasoning to find better plans for a river-crossing domain.

CIRCA Planning

We have extended the CIRCA planning and execution system (Musliner, Durfee, and Shin 1995; Musliner et al. 2008) to include a novel creative problem solving component that uses analogical reasoning to synthesize new plan operators by reasoning about resource affordances.

CIRCA has a higher-level "Adaptive Mission Planner" (AMP) that manages planning across an extended mission, selecting which goals are planned for during different mission phases and adapting the planning problems as needed to balance goal achievement against robustness (*e.g.*, against uncontrollable adversaries like weather) and mission time. The AMP repeatedly tasks a lower-level planner, the Controller Synthesis Module (CSM), with planning problems that are tailored to each mission phase. Like other symbolic Artificial Intelligence (AI) planners in the STRIPS tradition (Fikes and Nilsson 1971), CIRCA represents each world state as a finite set of features with discrete values drawn from finite domains (Goldman et al. 1997). Unlike most symbolic AI planners, CIRCA reasons about uncontrolled transitions (adversaries), temporal deadlines, possible failures, formally-verified safety-preserving plans, and real-time execution, but these concerns are not central to our approach to creative problem solving. We started with CIRCA because it already includes the meta-reasoning capability to reformulate problems for its domain-level planner, the CSM.

A CIRCA planning problem, configured by the AMP and posed to the CSM, includes initial conditions, controlled actions, uncontrolled transitions, goals (partial state descriptions), and a distinguished failure feature that must be avoided. It can also include numeric rewards for goal conditions. The CSM builds reactive control plans by anticipating possible future world states and selecting action choices for each state, including responding to unexpected or adversarial events and outcomes.

The CSM can operate in several different planning modes, depending on what kind of domain models are available and what level of reasoning about plan guarantees and performance is desired. "Classic" CIRCA uses models with unquantified nondeterminism (*e.g.*, actions may have diverse outcomes and the planner must reason about each of them, since it cannot control which outcome will occur). The CSM uses a formal verifier to prove that its plans achieve the goals and avoid catastrophic failures. In other modes, the model can include probabilistic transitions that have outcomes with known probability distributions, and goals can have associated rewards that allow tradeoff reasoning. We are using those models for this project, and running the CSM in its Maximize Expected Utility (MEU) mode.

In MEU mode, the CSM uses its usual heuristics to build reactive plans and then assesses their expected utility (EU—roughly, the sum of goal rewards times their probability of being achieved[1]). The CSM then keeps revising its plan, building new controller versions and checking to see if their EU is greater than the best plan so far. This cycle continues until a desired EU is reached, a timeout is reached, a desired number of controllers have been generated and evaluated, or every possible controller has been evaluated. Exhaustive search is only possible for small problems, due to combinatorial explosion. In practice, for larger problems, the CSM samples the space of possible controllers using heuristic search.

Illustrative Domain

As an example of the types of domains that can be solved using the CPS approach to analogical generalization, we have developed several variations of a "Dukes of Hazzard" (DoH) domain where an autonomous vehicle is trying to travel from East to West across a river canyon (see Figure 1). There are two possible routes across the river: a bridge which may or may not be destroyed, and a narrow spot in the canyon which may be jumped by the vehicle using a ramp. MEU mode allows the CSM to reason about both risk and reward, seeking the best balance. In the simplest DoH scenario, the balance is deliberately made obvious–the bridge is certain to achieve the goal of getting across the river, as long as it doesn't collapse (or get destroyed by an uncontrollable event, such as a flood) before the crossing is complete. The ramp jump is only somewhat likely to succeed, and its alternate outcome is catastrophic failure, so the planner will always prefer to use the bridge if possible. If the bridge is destroyed before the vehicle arrives, there's no choice but to try the ramp jump. However, in other domain variations, the bridge route may be less attractive. In some, there may be a high likelihood that the bridge will be destroyed while the car travels towards it, and in others the bridge may be defined as impassable before the scenario begins. In these cases, CPS will find the expected payoff of jumping the river more attractive.

To jump the river, CPS must apply the `jump_ramp` operator (see Figure 3). `jump_ramp` has two preconditions that must satisfied: a precondition on the value of `location`, which can be satisfied by a sequence of movement actions, and a `(have_ramp t)` precondition representing the presence of a "ramp" resource necessary to perform the action. As described below, CPS includes a notion of action-enabling resources, of which the ramp is an example. If there is no ramp available in the scenario, CPS must plan some way to obtain one or reason about substituting an available resource. That is where analogical reasoning for plan generalization comes in.

3 Creative CIRCA

Our Creative CIRCA architecture adds three broad new capabilities to the established CIRCA planning system (illustrated in Figure 2).

- First, we have introduced a simple, abstract notion of a "resource" to CIRCA planning. By distinguishing resources from other symbolic domain features, we enable new methods of reasoning that are specific to resources.
- Second, we have developed a method for *resource impasse detection*, using the planner's heuristic function to identify situations in which the lack of a resource defeats a potential problem solution.

[1]The actual EU calculation is more complicated because we allow different kinds of goals that get reward in different ways. Achievement goals get reward only once, repeated-achievement goals get reward every time they are re-achieved, and maintenance goals accumulate more reward the longer the system is expected to stay in a satisfying state.

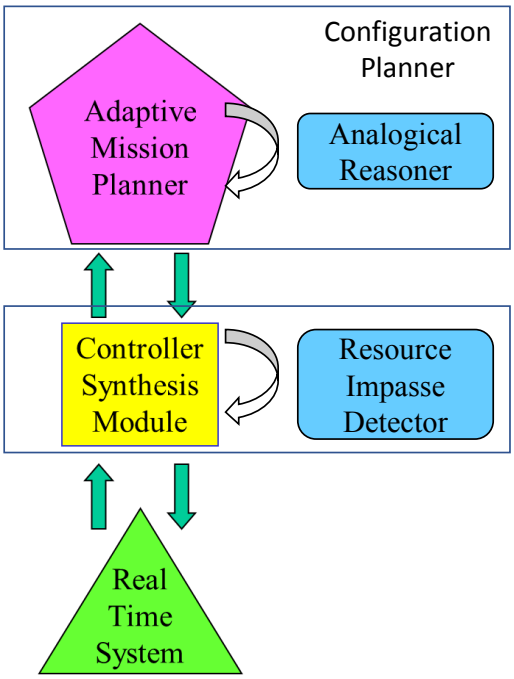

Figure 2: CPS enhances the existing CIRCA architecture with an analogical reasoning component and a *resource impasse detector*.

- Third, we have integrated an analogical reasoning system (AR) that uses semantic graph representations of resource affordances to propose possible resource substitutions, enabling the CP to generate new plan operators and creatively transform the problem considered by the CSM.

Resource Features

To support creative problem solving to overcome the absence of critical resources, we have enhanced the CIRCA world model with a representation of a *resource* feature. In our initial implementation, a resource can be any physical object or substance that is required to initiate a controlled action by the agent. Thus resources are closely related to the general notion of tools. Resources appear in the planning system's world model as special boolean resource features of the form `have_<resource_name>`. They also appear in a special database in the CP as named records with corresponding descriptions in the form of graph structures. For example, if the DoH car decides to jump over a river, it must meet the precondition `(have_ramp t)` (see Figure 3) and there will be a graph structure in the CP's data set that includes a description of a ramp (see Figure 5). These new domain modeling elements support new reasoning methods, as follows.

Resource Impasse Detection

To identify opportunities to use analogical reasoning to overcome planning challenges, CPS must recognize planning situations where a potentially useful action is not enabled. CPS

```
(def-action jump_ramp
  :preconds ((location ramp_trail_1) ;; and
             (have_ramp t))
  :postconds ((.90 (location safely_across_ramp) (canyonside west))
              ;; or
              (.10 (failure t))) ;; catastrophe!
  :wcet 10
  :delay-distribution (constant-distribution 10))
```

Figure 3: The `jump_ramp` action representation includes a "ramp" resource in its preconditions, and has probabilistic postconditions.

```
(def-action cross_bridge
  :preconds ((location bridge_trail_2) ;; and
             (bridge_condition open))
  :postconds ((location safely_across_bridge) ;; and
              (canyonside west))
  :wcet 10
  :delay-distribution (constant-distribution 10))
```

Figure 4: The `cross_bridge` action has a single set of deterministic postconditions.

exploits the structure of its heuristic graph to find such actions. During action choice, CPS builds a plan-graph style heuristic graph, building from individual goal propositions down through the domain transitions that can establish them (while ignoring the delete effects of transitions). Conjunctive preconditions of transitions are split into their individual propositions as subgoals, which are then supported in the same manner. For example, if `(canyonside west)` is a goal proposition, the heuristic graph will contain a root node for `(canyonside west)` with child nodes for each transition that establishes it. In turn, the children of those nodes are nodes for the preconditions of those transitions. When the heuristic graph has been completed, any leaf proposition nodes represent the unmet preconditions of possibly useful transitions. At that point, those nodes can be stashed for analysis, if there is reason to pursue an improvement to the first plan generated.

In the CPS architecture, it is the job of the CP to assess the quality of the plan returned by the CSM and decide whether to invest in the analysis of the collected heuristic leaf nodes. In the case that the CSM does not find any plan to reach the goal state, it is trivial to decide to re-plan. In other cases, the CP must assess some measure of plan quality. CIRCA supports a variety of different measures of plan quality including likelihood of failure, number of possibly reachable goals, and the estimated utility of the plan. In each case, an acceptable threshold for plan quality can be supplied as a parameter to the planner.

For example, in the DoH domain, if the bridge condition doesn't permit the `cross_bridge` operator (see Figure 4) and `have_ramp` is not `t`, the CSM will not be able to find *any* plan that achieves the goal of crossing the river. In that case, the CP will analyze the collected heuristic leaf nodes. Among the leaf nodes containing unsatisfiable action preconditions will be some for `(have_ramp t)` and some for `(bridge_condition open)`. At this point, the CP will have identified an impasse in planning that can be relieved by establishing an action precondition, but it has no possible actions that can directly achieve either condition.

The standardized resource naming allows the system to

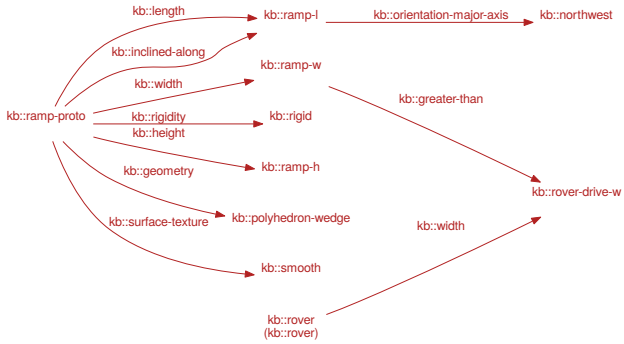

Figure 5: The CP's resource database contains a graph structure defining the ramp resource.

recognize that the ramp is a resource, and as such it may be able to generalize the `jump_ramp` action to use an alternative resource, whereas it has no way to generalize `cross_bridge` (in this particular domain version; one can easily imagine extended domains where the bridge is also modeled as a resource).

Analogical Reasoner

When the CPS identifies an opportunity to overcome an impasse caused by a missing resource, the AR searches for the best available substitute. To do so, the AR uses semantic structure-mapping (Friedman et al. 2017; McLure, Friedman, and Forbus 2015) to identify a candidate resource that is available, and synthesizes a new action operator.

CPS's structure-mapping component computes an approximate *maximal common edge subgraph* (MCES) using a greedy algorithm to avoid the NP-hard time complexity of identifying the optimal solution. Each MCES result contains the following information:

1. The *correspondences* of nodes and edges that map across the two semantic graphs. This describes the relational structure and parameters that the two semantic graphs (*i.e.*, resources in the planning problem) have in common.
2. The *graph complement*, which is the structure from each graph that does *not* correspond to the other.
3. A numerical *similarity score* derived from the cardinality of MCES correspondences.

When a missing resource has been identified, the AR generates a representation of the missing resource and uses structure-mapping to map this against representations of available resources. As described above, this describes what each available resource has in common (and in contrast) to the desired resource. Given multiple results from structure-mapping, AR presently ranks them by the size of the MCES correspondences (*i.e.*, where the desired resource corresponds maximally onto the available resource, with the smallest unmatched graph-complement).

As shown in Figure 5, the graph representations include attribute edges directed to associated values. For example, the ideal ramp has a `width` (internally represented

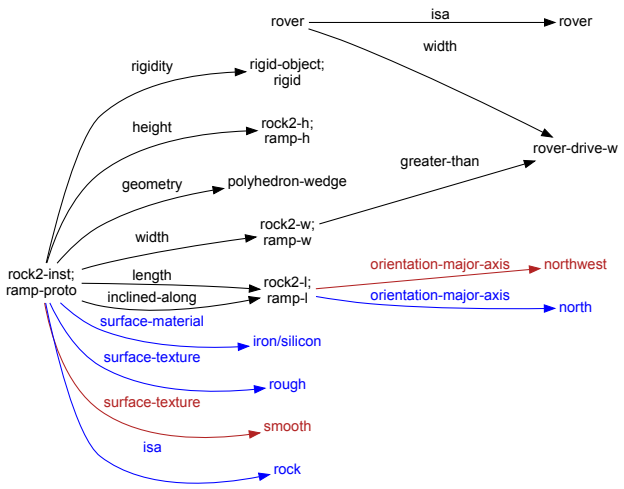

Figure 6: `rock-2`'s description graph differs from the ideal ramp in orientation and surface-texture, but matches most attribute edges.

as `ramp-w`) that is greater than the width of the rover (`rover-drive-w`).

In the DoH domain, we model the domain as including a variety of different objects, including several large rocks that have some properties that resemble a ramp. For example, Figure 6 shows that the `rock-2` graph differs from the ideal ramp graph in two attributes: orientation and surface texture. It also has superfluous `surface-material` and `isa` attributes.

This close similarity acts as a suggestion that `rock-2` might be a good substitute for the hoped-for ramp, particularly if those attributes could be adjusted.

4 Results

CPS's analogical plan generalization plays out in the following way in the DoH domain: suppose the planner has all of the model components associated with the action of jumping over the ramp, except that in the initial conditions there is no ramp present. The CSM can still build a good plan: just use the bridge. However, while building that plan it recognizes that there are states where it is not finding a way to achieve the goal (the states where the bridge has been destroyed). In those states, the CSM's heuristic graph can see that the `jump-ramp` action has desirable postconditions that would achieve the goal, but one key precondition is not satisfied: `(have-ramp t)`. The CSM records this as a "missing-resource" impasse.

When the CSM finishes with its best plan, the CP observes the missing-resource impasse annotation and calls out to its new AR component to see if a substitute can be found for the ramp. In our proof of concept, the AR uses graph matching over semantic graphs describing both prototypical desired objects (*e.g.*, a ramp) and object instances that are in the scenario (*e.g.*, several rocks, solar panels, the vehicle). The graph matching identifies ways in which instances do and do not match the desired prototype (*e.g.*, a specific rock matches some of the prototypical ramp's abstract, qualitative physical attributes such as rigidity, but lacks a smooth surface and is oriented the wrong way).

Selecting the best matching object (rock2), the AR interface creates a new "hypothesized" action for the CSM to plan with, via lexical substitution (*i.e.*, `jump_rock2`, with suitably modified pre- and post-conditions). The new action may also be given some additional uncertainty in its successful outcome probability, corresponding to the notion that this action has not been tried and thus may not be accurately modeled. The revised operator is given to the CSM, which replans and decides to try the new operator in the bridge-destroyed state, since it has no higher-EU options.

One advantage of this approach, using the planner to decide how to use hypothesized actions, is that it can automatically decide whether to do "trial runs" or "experiments" with a new action, before making commitments. In the `jump_rock2` example, the system risks catastrophic failure and may also complete the jump in a damaged but not destroyed state (wherein it would no longer be able to try again). Taking those risks into account, the planner might not plan to do a trial run, but commit on its first try. In a different scenario, with less risk of damage or complete failure, the system might decide to test the operator and examine its outcome. For example, if the problem is to hammer in a nail (Musliner 1994) and no hammer is present, but rocks are, the analogical reasoning could recognize that rocks share key relevant features with hammers (dense, heavy, hard surface) and propose a `hit_nail_with_rock` action. Since the new action would not threaten the agent itself, the planner might decide to try it out, see whether the nail moved, and only continue using that action if it was successful.

5 Evaluation

Using our CIRCA-based CPS implementation as a prototype and guide, we have designed a suite of scalability metrics for some aspects of Creative Problem Solving. We have started to build a scalable domain generator that can create different domains in order to assess CPS performance across different dimensions of variability. In the course of implementation, we have, of course, found opportunities to improve our CPS prototype and have continued to make it more robust and capable.

Building on our analogical examples where the system substitutes a rock for a ramp (or a rock for a hammer), we identified four important dimensions of scalability for our initial focus:

- The number of missing resources to overcome (by using a more-or-less suitable substitute).
- The size of the population of substitute candidates.
- The proportion of good (workable) candidates in the population.
- The richness of candidate descriptions.

To vary the number of missing resources to overcome, we create a set of domain models of fixed complexity (in terms of traditional planning metrics, such as number of operators) and increase the number of critical (required) resources that

are removed, in different versions of the problem. The CPS system must respond by identifying substitute resources from candidates in its environment. The characteristics of that candidate population can also be varied. First, we can change the absolute size of the overall population of candidates, challenging the system's efficiency in assessing substitutes. Second, we can vary the proportion of satisfactory candidates in the population. We expect that, depending on the ratio of satisfactory to total candidates, various heuristic approaches would be better or worse (*e.g.*, if the ratio is very high and experimenting with candidates is safe and low-cost, choosing randomly and experimenting might be a better approach than thinking hard about each individual candidate). Finally, we can vary the number of attributes used to describe the candidates to the CPS system, which should challenge the efficiency of the analogical matching system.

Leveraging our existing tooling for creating scalable domains along traditional AI planning dimensions, we have created a first version of this scalable domain generator and used our CPS system to solve many individual instances of the generated domains. The complexity of the generic domain is currently controlled by five parameters, each based on the dimensions of scalability mentioned above. The first three are the number of actions in the solution plan, n, the number of resources (a.k.a. tools) required to perform each action, r, and the number of missing tools, m. The remaining two parameters are the number of attributes that describe a resource/tool, a, and the number of possible values that can be assigned to an attribute, v. So by description alone, there are up to v^a unique tools that can exist. A given domain instance does not include all those tools. We enforce a *worst-case assumption* that no two actions use the same tool, so a generated domain has $nr - m$ tools available.

Domains are currently generated randomly given these parameters, which means it is unlikely that any of the m missing tools has a duplicate available under a different label. Analogical reasoning is thus required to assess which available tools are similar enough for substitution. We examined how just the analogical reasoner's performance scales with respect to changes in a, v, and $nr - m$ in Figure 7. Since the current attribute space is "flat" (only one predicate deep, rather than a more complex semantic graph), the analogical reasoning problem is quite simple and really just a vector comparison. Future work will extend the abstract tool representations to include more complex semantic relations.

The full CIRCA-based solver must call the analogical reasoner at least m times as it reasons over viable tool substitutions; its runtime changes with respect to changes in r and m are shown in Figure 8.

These performance results suggest that, even without optimization, the domain-independent solving algorithms are working well; effort scales appropriately with domain size, and the planner is not performing any backtracking or search in these simple domains. We have already identified additional improvements to the domain-independent reasoning that should allow the system to reason even more quickly about multiple missing tools.

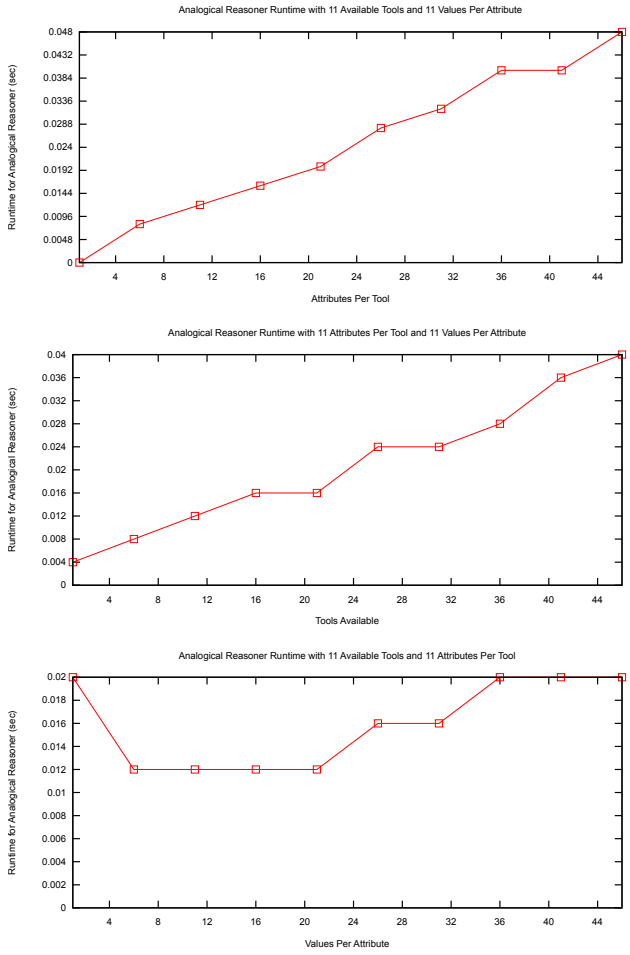

Figure 7: Performance of the Analogical Reasoner on a Generic Domain

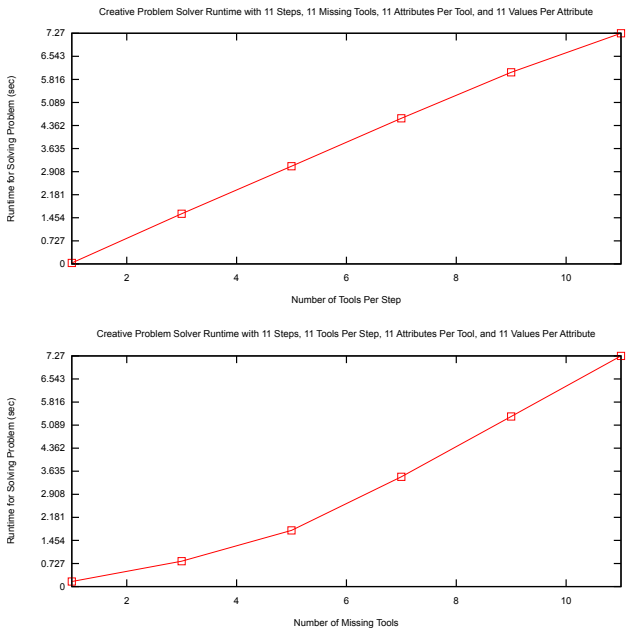

Figure 8: Performance of the CIRCA-based Creative Problem Solver on a Generic Domain

6 Related Work

Although there is a reasonable body of literature studying artificial intelligence for creativity with respect to generating novel content, CPS approaches creativity with respect to less-obvious decision making strategies for solving problems. Sarathy and Scheutz (2018) outline the basic elements of agents that exhibit creative problem solving strategies, naming the broad class of challenges as *MacGyver Problems*. Not to be confused with a focus on finding ways to attach tools together to generate new ones with alternative functionality (Nair et al. 2019), MacGyver Problems refer to problems where the subset of the world's state space that is reachable by an agent's current knowledge and abilities does not include the goal state(s). Thus the agent must rely on a set of techniques and strategies that can discover ways to expand its reachable state space, which should eventually make it possible to solve the task and reach a goal state. Sarathy (2018) has also begun to explore possible approaches humans use based on existing neuroscience literature.

The planning and scheduling community has also investigated replacing elements of a planning problem's definition to address tasks. Also investigating cases where problems are not solvable, Göbelbecker et al. (2010) identified possible modifications to the initial state that could render the problem solvable. These modifications provided *excuses* to justify the unsolvability, which serves explainable AI planning (Hoffmann and Magazzeni 2019) more than the actual problem-solving aspect unless the planning agent is able to use the excuse to render the problem solvable. In the case where one observes a planning agent whose model is unknown to the observer, Aineto et al. (2019) define the model recognition problem. Their definition assumes that fluents composing the state space are the same for both the planning agent and observer, but the observed actions are only identified by name such that their implementation (preconditions and effects) needs to be selected from a set of hypothesized implementations. Their approach for solving model recognition involves compiling a new planning task whose actions include editing preconditions and effects of the named action definitions as well as executing the modified actions.

Our approach for expanding the set of reachable states instead involves generating new action groundings by finding viable object substitutions that seem analogically similar. Analogical reasoning identifies similarities between objects with respect to their qualities, which has been used to model consistency between similar-yet-different things. For example, interfaces for various computer applications (Rieman et al. 1994), even though they are tools for different types of tasks, have common features such as opening and saving files. These similar functionalities are often represented using similar buttons and placement so that users can more easily learn how to use a new application based on their previous interface experiences. A number of analogical reasoning techniques in artificial intelligence have been based on structure-mapping theory (Gentner 1983), including the approach we used in our implementation of CPS (Friedman et al. 2017; McLure, Friedman, and Forbus 2015) as well as analogy ontologies that use first-principles reasoning (Forbus, Mostek, and Ferguson 2002).

7 Conclusions and Future Work

When resources and tools are identified by name or label in a domain, a planning agent may fail to solve a problem when a specific tool or resource, required in its planning model, is not available. However, the resource or tool is usually necessary for some functional purpose or property rather than for its specific name/label alone. Analogical reasoning enables agents to reason about these properties and identify other tools or resources that are available and may serve as a substitute to find an alternative solution. This aspect of creative problem solving allows agents to generalize their ability to plan, by extending the set of reachable states via repurposing of objects in the environment to replace those that are needed, but not present.

We introduced our integration of an analogical reasoning system into the CIRCA planning architecture, illustrated how it can creatively solve problems with missing resources, and explored how various factors affect the scalability of both the analogical reasoner and the planner. Besides continuing to investigate other factors for scaling and their impacts on the system's performance, we see some important areas for future work.

Constraining the analogical reasoning: We have shown that semantic structural similarity is a heuristic for generalizability, but specific dimensions, features, or capabilities of a resource, such as the affordances of the desired resource with respect to the operator, are paramount considerations. Consequently, one direction for future work is imposing constraints on the analogical reasoning to prioritize

and filter its solutions to identify the elements with the desired affordances.

Experimentation and empirical feedback: Enabling experimentation to identify and/or verify how to use the analogically-similar resource for the desired purpose is a valuable direction for future work. Specifically, positive or negative feedback might be incorporated to broaden the generalization even further (if successful), or avoid making future mistakes (in the case of failure). Furthermore, since analogical generalization changes the planner's configuration, incorporating negative feedback could help prevent introducing error into the configuration.

Expanding metrics: We currently assess CPS's performance with respect to runtime based on scaling factors. However, alternative plans can have varying degrees of creativity. For example, replacing a missing ramp resource with another ramp resource that is available nearby might not seem as creative as replacing it with a rock that has the needed ramp affordances Sarathy and Scheutz. (2018) proposed some metrics conceptually, such as the number of changes to the reachability over the state space, but these focus on the creativity of the problem solving agent rather than the creativity of the alternative plan(s). Exploring ways to measure both creative problem solving agents and their artifacts could present insights into creative problem solving as well as identify novel ways to generalize planning systems and plans when presented with novel domains and scenarios.

8 Acknowledgments

This material is based upon work supported by DARPA under Contract No. W31P4Q-18-C-0064. Distribution Statement 'A' (Approved for Public Release, Distribution Unlimited). DISCLAIMER: The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either express or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

References

Aineto, D.; Jiménez, S.; Onaindia, E.; and Ramírez, M. 2019. Model recognition as planning. In *Proceedings of the Twenty-Ninth International Conference on Automated Planning and Scheduling*, ICAPS'19, 13—-21. Berkeley, CA, USA: AAAI Press.

Fikes, R. E., and Nilsson, N. J. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2:189–208.

Forbus, K. D.; Mostek, T.; and Ferguson, R. 2002. An analogy ontology for integrating analogical processing and first-principles reasoning. In *Proceedings of the Fourteenth Conference on Innovative Applications of Artificial Intelligence*, IAAI'02, 878–885. Edmonton, Alberta, Canada: AAAI Press.

Friedman, S. E.; Burstein, M. H.; Rye, J. M.; and Kuter, U. 2017. Analogical localization: Flexible plan execution in open worlds. In *ICCBR (Workshops)*, 33–42.

Gentner, D. 1983. Structure-mapping: A theoretical framework for analogy. *Cognitive Science* 7(2):155–170.

Göbelbecker, M.; Keller, T.; Eyerich, P.; Brenner, M.; and Nebel, B. 2010. Coming up with good excuses: What to do when no plan can be found. In *Proceedings of the Twentieth International Conference on Automated Planning and Scheduling*, ICAPS'10, 81—-88. Toronto, Ontario, Canada: AAAI Press.

Goldman, R. P.; Musliner, D. J.; Boddy, M. S.; and Krebsbach, K. D. 1997. The CIRCA model of planning and execution. In *Working Notes of the AAAI Workshop on Robots, Softbots, Immobots: Theories of Action, Planning and Control*.

Goldman, R. P.; Pelican, M. J. S.; and Musliner, D. J. 2004. Guiding planner backjumping using verifier traces. In *Proc. Int'l Conf. on Automated Planning and Scheduling*.

Hoffmann, J., and Magazzeni, D. 2019. *Explainable AI Planning (XAIP): Overview and the Case of Contrastive Explanation (Extended Abstract)*. Bolzano, Italy: Springer International Publishing. 277–282.

McLure, M. D.; Friedman, S. E.; and Forbus, K. D. 2015. Extending analogical generalization with near-misses. In *AAAI*, 565–571.

Musliner, D. J.; Pelican, M. J. S.; Goldman, R. P.; Krebsbach, K. D.; and Durfee, E. H. 2008. The evolution of circa, a theory-based ai architecture with real-time performance guarantees. In *AAAI Spring Symposium on Architectures for Intelligent Theory-Based Agents*.

Musliner, D. J.; Durfee, E. H.; and Shin, K. G. 1993. CIRCA: a cooperative intelligent real-time control architecture. *IEEETSMC* 23(6):1561–1574.

Musliner, D. J.; Durfee, E. H.; and Shin, K. G. 1995. World modeling for the dynamic construction of real-time control plans. *Artificial Intelligence* 74(1):83–127.

Musliner, D. J. 1994. Using abstraction and nondeterminism to plan reaction loops. In *Proc. National Conf. on Artificial Intelligence*, 1036–1041.

Nair, L. V.; Srikanth, N. S.; Erikson, Z.; and Chernova, S. 2019. Autonomous tool construction using part shape and attachment prediction. In *Proceedings of Robotics: Science and Systems*, 1–10.

Rieman, J.; Lewis, C.; Young, R. M.; and Polson, P. G. 1994. Why is a raven like a writing desk?: Lessons in interface consistency and analogical reasoning from two cognitive architectures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, 438–444. Boston, Massachusetts, USA: ACM.

Sarathy, V., and Scheutz, M. 2018. MacGyver problems: AI challenges for testing resourcefulness and creativity. *Advances in Cognitive Systems* 6:1–15.

Sarathy, V. 2018. Real world problem-solving. *Frontiers in Human Neuroscience* 12:1–14.