Deceptive and Counter-Deceptive Machines
Papers from the AAAI 2015 Fall Symposium

Formalizing Deceptive Reasoning in *Breaking Bad:* Default Reasoning in a Doxastic Logic

John Licato
licatoj@ipfw.edu
Analogical Constructivism and Reasoning Lab (ACoRL)
Indiana University and Purdue University-Fort Wayne

Abstract

The rich expressivity provided by the cognitive event calculus ($\mathcal{CEC}$) knowledge representation framework allows for reasoning over deeply nested beliefs, desires, intentions, and so on. I put $\mathcal{CEC}$ to the test by attempting to model the complex reasoning and deceptive planning used in an episode of the popular television show *Breaking Bad*. $\mathcal{CEC}$ is used to represent the knowledge used by reasoners coming up with plans like the ones devised by the fictional characters I describe. However, it becomes clear that a form of nonmonotonic reasoning is necessary—specifically so that an agent can reason about the nonmonotonic beliefs of another agent. I show how $\mathcal{CEC}$ can be augmented to have this ability, and then provide examples detailing how my proposed augmentation enables much of the reasoning used by agents such as the *Breaking Bad* characters. I close by discussing what sort of reasoning tool would be necessary to implement such nonmonotonic reasoning.

An old joke, said to be a favorite of Sigmund Freud, opens with two passengers, Trofim and Pavel, on a train leaving Moscow. Trofim begins by confronting Pavel, demanding to know where he is going.

> **Pavel**: "To Pinsk."
> **Trofim**: "Liar! You say you are going to Pinsk in order to make me believe you are going to Minsk. But I know you are going to Pinsk!" (Cohen 2002)

Fictional stories can sometimes capture aspects of deception in the real world, especially between individuals who are skilled at reasoning over the beliefs of others (second-order beliefs), the beliefs of one party about the beliefs of another (third-order beliefs), and so on. For example, an agent a desiring to deceive agent b may need to take into account agent b's counter-deception measures (where the latter measures may be directed back at agent a, as was suspected by poor Trofim). Such fictional stories may thus sometimes be a suitable source of test cases for frameworks specializing in the representation of, and reasoning over, complex doxastic statements. The cognitive event calculus ($\mathcal{CEC}$) promises to be such a framework, given its ability to represent beliefs, knowledge, intentions, and desires over time (Arkoudas and Bringsjord 2009).

In this paper, I will attempt to model the reasoning used by agents in an episode of the television series *Breaking Bad*. Episode 13 of season 5, entitled *To'hajiilee*, is notably rich in deceptive behaviors between characters, being a point in the series' overall story arc where the conflict between several consistently wily characters comes to a climax. One group (Jesse and Hank) devises a plan to lure, trap, and catch another character (Walt), and I try to answer two questions about their plan in this paper: First, what sort of reasoning and knowledge representation would be necessary to devise such a plan as the one created by Jesse and Hank? Second, is $\mathcal{CEC}$ sufficiently powerful to represent such knowledge and serve as a base framework for such reasoning?

Section 1 will argue that even an analysis of how well $\mathcal{CEC}$ can model reasoning in a fictional story can be beneficial to the field of automated human-level reasoning, discussing related literature. I give an overview of $\mathcal{CEC}$ in Section 2, followed by a synopsis of the relevant portions of *To'hajiilee*'s plot (Section 3.1). An analysis of the plan generation used by the characters[1] in Section 3.2 suggests the need for a form of nonmonotonic reasoning that requires, at a minimum, reasoning over second-order beliefs. I then spend some time explaining how this nonmonotonic reasoning can work in $\mathcal{CEC}$. The paper wraps up with a discussion of implications for the future of deceptive and counter-deceptive AI (Section 5).

1 Why Bother Modeling Reasoning in Plots?

The cognition of deception is particularly interesting to model: Knowing when to deceive in social situations may make for robots that are better accepted socially (Wagner and Arkin 2009; Sharkey and Sharkey 2011). Deceptive machines may indeed be the inevitable consequence, or perhaps explicit goal, of human-level AI (Castelfranchi 2000; Clark and Atkinson 2013).

Instances of deception in fiction are not difficult to find. Some variant of deceptive behavior seems to appear in any story involving characters containing beliefs, intentions, and desires about the beliefs of other characters, depending on

[1] Of course, the characters I discuss here are fictional. I really am talking about the work of the writers of the show, who are reasoning from the perspectives of the fictional characters. It will be more convenient in this paper to simply say it is the fictional characters doing the reasoning.

the definition of deception one accepts. Although some stories are better than others at accurately portraying realistic behaviors, all were written at some point by imaginative human beings (with some exceptions, cf. (Bringsjord and Ferrucci 1999)). They therefore offer clues about the human ability to think deceptively and counter deceptively; e.g., a plan of deception devised by a fictional character, at the very least, tells us what types of plans humans are capable of both comprehending (as the readers of a story do) and creatively generating (as the writers did). For researchers interested in understanding the expressivity of human-level thought, stories of deception are useful benchmarks.

2 An Overview of $\mathcal{CEC}$

The cognitive event calculus ($\mathcal{CEC}$) is a first-order modal logic for knowledge representation first introduced by Arkoudas and Bringsjord (2009) as a way to model Piaget's false-belief task. A member of the cognitive calculi family of logics (Bringsjord et al. 2015), $\mathcal{CEC}$ contains operators for several mental states and events: **B**elief, **K**nowledge, **I**ntention, **D**esire, **C**ommon knowledge, and **S**peech acts. Note that not all of these operators are introduced in Arkoudas and Bringsjord (2009); rather, much of the current version of $\mathcal{CEC}$ reflects subsequent developments, most of which were produced in parallel with work on the *deontic* cognitive event calculus ($\mathcal{DCEC}^*$), an extension of $\mathcal{CEC}$ (Bringsjord et al. 2014).

$\mathcal{CEC}$ is loosely based on the event calculus (Kowalski and Sergot 1986), but departs from it and other similar logics in several important ways, two of which are especially relevant to this paper's purposes:

- Although no formal semantics is fully defined for $\mathcal{CEC}$, there is a preference for proof-theoretic (and the highly related argument-theoretic) semantics and a natural deduction (Jaśkowski 1934) style of inference. Although there are some cases where cognitively implausible techniques such as resolution may assist in the proof-finding process, the underlying inferences are rooted in a set of constantly refined inference rules.
- $\mathcal{CEC}$ rejects the use of logical operators and inference rules in contexts for which they were not designed. Standard deontic logic (SDL) made the mistake of trying to define an obligation operator as a direct analog of the necessity operator from standard modal logic, with disastrous consequences (Chisholm 1963; McNamara 2014).

Most $\mathcal{CEC}$ formulae contain terms for at least one agent, a temporal unit, and a nested formula. For example, $\mathbf{B}(a,t,\phi)$ is read "agent a believes ϕ at time t". There are two exceptions to this form: $\mathbf{C}(t,\phi)$ is read "all agents believe ϕ at time t", and $\mathbf{S}(a,b,t,\phi)$ says "at time t, agent a says ϕ to agent b". The syntax used in this paper is pictured in Figure 1.

3 Can $\mathcal{CEC}$ Model Hank's Deceptive Reasoning?

The cognitive event calculus provides a formalism for representing cognitively rich knowledge, but as this section will show, an augmentation is needed before the sort of reasoning used by the *Breaking Bad* characters can be faithfully modeled.

Syntax

$$S ::= \text{Object} \mid \text{Agent} \mid \text{Self} \sqsubset \text{Agent} \mid \text{ActionType} \mid \text{Action} \sqsubseteq \text{Event} \mid \text{Moment} \mid \text{Boolean} \mid \text{Fluent} \mid \text{Numeric}$$

$$f ::= \begin{array}{l} action : \text{Agent} \times \text{ActionType} \rightarrow \text{Action} \\ initially : \text{Fluent} \rightarrow \text{Boolean} \\ holds : \text{Fluent} \times \text{Moment} \rightarrow \text{Boolean} \\ happens : \text{Event} \times \text{Moment} \rightarrow \text{Boolean} \\ clipped : \text{Moment} \times \text{Fluent} \times \text{Moment} \rightarrow Boolean \\ initiates : \text{Event} \times \text{Fluent} \times \text{Moment} \rightarrow \text{Boolean} \\ terminates : \text{Event} \times \text{Fluent} \times \text{Moment} \rightarrow \text{Boolean} \\ prior : \text{Moment} \times \text{Moment} \rightarrow \text{Boolean} \\ interval : \text{Moment} \times \text{Boolean} \\ payoff : \text{Agent} \times \text{ActionType} \times \text{Moment} \rightarrow \text{Numeric} \end{array}$$

$$t ::= x : S \mid c : S \mid f(t_1, \ldots, t_n)$$

$$\phi ::= \begin{array}{l} t : \text{Boolean} \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \\ \mathbf{P}(a,t,\phi) \mid \mathbf{K}(a,t,\phi) \mid \mathbf{C}(t,\phi) \mid \mathbf{S}(a,b,t,\phi) \mid \mathbf{S}(a,t,\phi) \\ \mathbf{B}(a,t,\phi) \mid \mathbf{D}(a,t,holds(f,t')) \mid \mathbf{I}(a,t,happens(action(a^*,\alpha),t')) \end{array}$$

Figure 1: The $\mathcal{CEC}$ Syntax Used in this Paper

3.1 Plots and Plans

Three characters are relevant to the plot at this point. Walter White ("Walt") is a now-retired methamphetamine kingpin who is in possession of over $80 million, trying to live the remainder of his life (likely not to be long due to a recurrence of cancer) in peace with his family. Henry Schrader ("Hank") is Walt's brother-in-law, but is also a special agent with the Drug Enforcement Agency while being completely unaware of Walt's second life as a drug dealer. Hank has been searching for the meth manufacturer known only as "Heisenberg," unaware that it is Walt. Finally, Jesse Pinkman is a former student of Walt's (when Walt was still working as a high school chemistry teacher) who Walt recruited to start and build his drug empire. Due primarily to selfish choices made by Walt over the course of their collaboration, Jesse's life of late has been beset by many tragedies, and Jesse wants revenge.

In the episodes leading up to *To'hajiilee*, several revelations are made. Hank begins to strongly suspect that Walt is Heisenberg, but realizes after a confrontation with Walt that he has no clear way to collect evidence that can convict him. When Walt confirms that Hank knows his secret, Walt immediately drives to a location in the desert and buries his money, not telling anyone where he has hidden it. Meanwhile, Jesse figures out some of the terrible things Walt has done and reluctantly decides to team up with Hank to ac-

quire evidence to put Walt away (a collaboration that Walt is unaware of, and believes is not even a distant possibility—this misplaced faith in Jesse proves to be a significant strategic disadvantage for Walt).

A plan devised by Jesse is only partially successful, as it provides Jesse and Hank with the knowledge that Walt's money is buried somewhere in the desert inside specific types of barrels, and that there are exactly seven such barrels, but the location of the barrels is still inaccessible. At this point (time t_0), Hank devises a plan to get Walt to unintentionally reveal the location of the money:

- Jesse and Hank will start by hiding at Walt's place of business (a car wash).
- When Walt arrives at the car wash (time t_1), Jesse is to send Walt a picture of a barrel, of the same sort Walt used to bury his money, filled with cash and looking like it was recently dug out of a hole in the desert. Jesse is to call Walt and tell him several things:
  - that Jesse knows the location of Walt's money and will destroy a certain amount every minute until Walt arrives at that location,
  - that Jesse knows there are six other barrels that Walt buried,
  - that Jesse discovered the location of Walt's money through an elaborate plan involving a GPS tracker on Walt's van that he was unaware of, and
  - that Jesse orders Walt not to hang up the phone and call for help, or he will instantly destroy the entire amount.
- As Walt is driving to the location of the money, Jesse and Hank will follow him covertly, while Jesse keeps Walt on the phone.
- When Walt finally arrives at the location of the money, (time t_2), Hank will arrest Walt and collect his money as evidence.

Hank's plan does not unfold as expected. Walt ends up admitting, on the phone, his role in many of his most horrendous crimes (providing evidence for Hank who is presumably recording the conversation), and upon arriving at the location of Walt's money, they encounter another hostile group, both possibilities that Hank could have never predicted. The success or failure of his plan, however, is not the focus of this paper. Instead, I indend to take steps towards understanding what sort of knowledge representation and reasoning would be required for an artificial agent to generate a plan with the sophistication and foresight of Hank's.

In $\mathcal{CEC}$, there are no constructs specifically created for representing plans, though this shortcoming can be easily remedied. Following Russell and Norvig (2010), an *action schema* consists of a precondition, an action, and an effect, along with a set of free variables. A *plan* is a sequence of grounded action schemas (action schemas whose free variables have all been instantiated) carried out in service of some goal, where the goal is usually one of the effects of the last action schema in the sequence.

As a first attempt at representing an action schema in $\mathcal{CEC}$, first let the formula $causes(holds(pre, t), happens(act, t), holds(eff, t'))$ hold if and only if: when precondition pre holds at time t, performing action act will cause effect eff to hold at time t', where $t' > t$. An action schema consists of a formula whose root predicate is $causes$ (so that it is of the form $causes(x, y, z)$ and (if it is ungrounded) a set of free variables $v_1, ..., v_n$.

The predicate $causes$ may seem to capture the naïve intuition behind action schemas. But in practice, actually using an action schema in an doxastic logic to infer $holds(eff, t')$ turns out to be more complex than the $causes$ predicate can handle in its present form. As I will explain next, the use of a technique like *default reasoning* is necessary.

3.2 The Need for Default Rules in $\mathcal{CEC}$

The process of plan generation might start with a desire for some event to occur or state to hold. That desire may translate into an extremely sparse high-level plan (e.g., "I plan to make what I desire true"), followed by iterative elaborations of that plan. Under this model, Hank would have started with the desire to convict and incarcerate Walt. One way to achieve this desire is to ensure that two conditions are met: First, that Hank is in possession of evidence sufficient to convict Walt; and second, that Hank has Walt in custody. The knowledge of such a connection is represented as an action schema, and Hank could presumably search backwards through these schemas to create a plan (similar to the process of explanation generation in (Licato, Sun, and Bringsjord 2014)).

But such action schemas would be lacking in even a slightly more robust simulation, much less the real world. Consider a simple example: Assume Hank believes that if Walt believes his money is in danger of being destroyed by Jesse (the precondition), and Hank takes the action of secretly following Walt (the action), then Walt will lead Hank directly to the money (the effect). But what if the effect of this action schema not hold, even when the precondition holds and the action takes place? Walt might take a detour and obtain something belonging to Jesse to hold hostage, or he might make a call to another character who would provide Walt with information leading him to doubt that Jesse actually knows where the money is. There are virtually an unlimited number of possibilities, or *confounders*, that might happen to prevent the effect from happening. It is the job of the planning agent to try to anticipate as many of these confounders as possible, or to make use of causal knowledge that is subject to as few confounders as possible.

Worse yet, if the action schemas are treated as simple material conditionals within $\mathcal{CEC}$, confounders may lead to logical contradictions. Consider the case where preconditions $p_1, ..., p_n$ and action α lead to effect e, and preconditions $p_1, ..., p_{n+1}$ and action α lead to effect $\neg e$. It is conceivable that both sets of preconditions are met (since one is a subset of the other), and a naïve approach to inferring the effects of action schemas might lead one to reason that both e and $\neg e$ hold, which clearly can not be the case. Furthermore, it is unrealistic and intractable to expect a formalization to populate all of its action schemas with *every possible* precondition, taking into account all possible confounders. This

is particularly true in a deception situation, where planning agents may need to consider other agents' first-order beliefs, second-order beliefs, and so on as possible confounders.

Given these difficulties, it seems reasoning in a counter-deceptive fashion is better captured by a form of nonmonotonic reasoning. Here I will draw on *default reasoning*, a form of nonmonotonic reasoning that allows action effects to be represented and reasoned over without having to list all of the possible confounders (Reiter 1980; Horty 2012; Koons 2014). A *default rule* is a triple $(p, \mathbf{J}, c)$ where p is a prerequisite, $\mathbf{J}$ is a set of justifications, and c is a conclusion. The default rule is understood as saying if p holds, and each of the $j \in \mathbf{J}$ are consistent with a pre-established background theory, then c can be inferred. For example, if $p = isBird(x)$ and $e = canFly(x)$, a useful justification would be $j_1 = canFly(x)$. In other words, if x is a bird and x being able to fly is not inconsistent with the background theory, then we can safely infer that x can fly. This ensures consistency with an instantiation of x as a penguin; if x is a penguin, since it is known that penguins cannot fly ($\neg canFly(penguin)$), the inference $canFly(penguin)$ is not made.

Drawing from default rules, a *default action schema* (DAS) shall be defined as a set of free variables $v_1, ..., v_n$ and a formula of the form:

DAS (Default Action Schema) Formula

$$causes_{_D}(holds(pre, t), happens(act, t), J, pri, holds(eff, t'))$$

Where:

- precondition pre is a $\mathcal{CEC}$ formula
- act is an action
- pri is a priority value used for prioritizing competing default action schemas
- eff is the effect.
- J, a $\mathcal{CEC}$ formula called the *justifications*, is of the form $j_1 \wedge ... \wedge j_m$ (each j_i will be referred to as a justification).

Just as producing inferences using default rules requires performing consistency checks relative to some background theory, the effects of DASes hold only if they are consistent with some background theory Π. The special function $\texttt{consistent}(\Pi, j)$ is new to $\mathcal{CEC}$, and it is meant to approximate the notion of consistency used in default reasoning. Π is a set of $\mathcal{CEC}$ formulae, and J is the justifications. An implementation of default action schemas will have to check if the justifications are consistent with Π, subject to implementation-specific practical considerations (for example, an implementation might try to prove $\Pi \cup \{J\} \vdash_{\mathcal{CEC}} \bot$ and regard the reaching of a pre-set time limit as sufficient to treat $\texttt{consistent}(\Pi, J)$ as true). Therefore, `consistent` should not be thought of as a normal function, but rather a special instruction to the reasoner to perform a consistency evaluation (it is for this reason and other reasons discussed in Section 3.3 that I write `consistent` in teletype font).

Making use of the `consistent` function allows the introduction of a new $\mathcal{CEC}$ inference rule, **DAS Extension** (Figure 2). The notation, Π_a^t is used to represent the background theory of agent a at time t. Section 3.3 will explain why evaluation of rule **DAS extension** requires special considerations.

If rule **DAS extension** is used to add an inference to a background theory, the resulting background theory is referred to as an *extension* of the previous theory (Koons 2014). As will be explained in Section 3.4, there is often a choice to make between which default action schemas to use to create inferences.

3.3 Reasoning Over DASes of Others

The introduction of default reasoning into $\mathcal{CEC}$ becomes more powerful when default reasoning constructs become objects over which agents can reason. For example, when Hank is generating his plan to deceive Walt, he needs to anticipate how actions made as part of his plan will affect Walt. Hank may need to modify his plans if he believes Walt will act or generate beliefs in accordance with a certain DAS, even if that DAS is not one that Hank himself is subject to.

At time t_0, Hank knows that there was no GPS tracker on the van Walt used when burying his money. But Hank also knows that Walt does not know whether the van did or did not have GPS. Hank can therefore justifiably believe that Walt, after being told that Jesse knows how many barrels were buried, will conclude by default that the van did in fact have a GPS tracker. Hank's belief can be captured by the formula:

$$\begin{aligned}&\mathbf{B}(hank, t_0, \mathbf{B}(walt, t_0, causes_{_D}(\\ &\quad holds(\mathbf{B}(walt, t_0, \mathbf{B}(jesse, t_0, numBarrels(6))), t_0),\\ &\quad happens(\mathbf{S}(jesse, walt, t_0, hasGPS(van)), t_0),\\ &\quad \mathbf{B}(walt, t_0, hasGPS(van)),\\ &\quad 3,\\ &\quad holds(\mathbf{B}(walt, t_0, hasGPS(van)), t_0))))\end{aligned}$$

Hank must therefore evaluate a DAS from Walt's perspective. According to rule **DAS extension**, this requires evaluation of the `consistent` function. However, because the call to the `consistent` function is within the scope of a $\mathbf{B}$ operator, any evaluation of consistency must be done from *the perspective of the agent doing the reasoning*. Π_a^t, in the case of the formula $\mathbf{B}(a, t, \texttt{consistent}(\Pi_a^t, J))$, consists of all formulas of the form $\mathbf{B}(a, t, \phi)$. But what about cases where an agent is reasoning about the DASes of another agent? In Hank's case, he needs to perform a consistency check of justifications with the background theory of Walt ($\Pi_{Walt}^{t_0}$). But Hank would not evaluate the DAS against Walt's actual background theory, because Walt may have beliefs that Hank is not aware of. Rather, Hank should be evaluating the DAS against *what Hank believes Walt's background theory is*!

My proposed solution is as follows: evaluation of the truth value of the `consistent` function should take into account the entirety of the nested formula in which the function occurs. For instance, imagine we are faced with a formula such as:

Rule DAS extension:

$$\frac{\mathbf{B}(a, t, causes_{_D}(holds(pre, t), happens(act, t), J, pri, holds(eff, t'))), t \leq t'}{\mathbf{B}(a, t, \texttt{consistent}(\Pi^t_a, J)) \rightarrow \mathbf{B}(a, t, holds(eff, t'))}$$

Figure 2: The Rule "DAS Extension"

$$\mathbf{B}(a, t, \mathbf{B}(b, t, \texttt{consistent}(\Pi^t_b, J)))$$

In this formula, the evaluation of $\texttt{consistent}(\Pi^t_b, J)$ must be consistent with what agent a thinks the background theory of agent b is, i.e. all formulas of the form $\mathbf{B}(a, t, \mathbf{B}(b, t, \phi))$. This example can be stated more generally:

Evaluation of `consistent`

For any formula of the form:

$$\mathbf{B}(a_1, t, \mathbf{B}(a_2, t, ...\mathbf{B}(a_n, t, \texttt{consistent}(\Pi^t_{a_n}, J)))...)$$

The evaluation of $\texttt{consistent}(\Pi^t_{a_n}, J)$ will return true if and only if J cannot be shown to be inconsistent with the set consisting of all formulas of the form $\mathbf{B}(a_1, t, \mathbf{B}(a_2, t, ..., \mathbf{B}(a_n, t, \phi))...)$ (ranging over all possible ϕ).

3.4 Using Default Action Schemas to Pre-empt Confounders

In situations of deception and counter-deception, an agent can make use of DASes to search for possible confounding factors in a guided way. The basic idea is simple: You can start with a simple plan, and search for possible confounders to that plan. If a confounder is discovered, then the plan is augmented to ensure the goals of the plan are still satisfied. This is repeated iteratively until the reasoner is satisfied.

Drawing another example from the *Breaking Bad* plot, Hank may start planning by deciding that he will discreetly follow Walt, having reasoned from a DAS that says if Walt believes Jesse knows where his money is, and Jesse convinces Walt he will destroy the money, then Walt will head directly to the money's location, accompanied by nobody. But there are a virtually unlimited number of confounders, e.g., Walt may decide, while en route to the money, to call someone else to form a counter-strategy. In such a case, Walt likely will not come alone; worse yet, he may not go to the money's location at all. Hank must therefore reason over the following DASes:

$$\begin{aligned}
H_1 : causes_{_D}(& \\
& holds(\mathbf{B}(walt, t_0, knowsLoc(jesse, money)), t_0), \\
& happens(\mathbf{S}(jesse, walt, t_0, \\
& \quad \mathbf{I}(jesse, t_1, destroy(money))), t_0), \\
& holds(travelAlone(walt, location(money)), t_0), \\
& 1, \\
& holds(travelAlone(walt, location(money)), t_0))
\end{aligned}$$

$$\begin{aligned}
H_2 : causes_{_D}(& \\
& holds(\mathbf{B}(walt, t_0, knowsLoc(jesse, money)), t_0) \wedge \\
& \quad happens(callsForHelp(walt), t_0), \\
& happens(\mathbf{S}(jesse, walt, t_0, \\
& \quad \mathbf{I}(jesse, t_1, destroy(money))), t_0), \\
& \neg holds(travelAlone(walt, location(money)), t_0), \\
& 2, \\
& \neg holds(travelAlone(walt, location(money)), t_0))
\end{aligned}$$

DASes therefore provide a structured way to search through possible confounders to a plan. In this case, the confounder can be pre-empted by Hank, by having Jesse threaten to burn all of the money if Walt hangs up the phone or calls for help (a detail that was indeed part of Hank's final plan).

DASes H_1 and H_2, are two DASes that may have their preconditions and actions satisfied simultaneously, yet they produce incompatible conclusions. Such a situation is similar to the *Nixon diamond*, a situation in which two default rules can produce incompatible inferences (Horty 2012). In the Nixon diamond, the two default rules are essentially that Quakers are pacifist, and Republicans are not pacifist. Nixon, however, is both a Quaker and a Republican. Is he or is he not pacifist?

The priority value pri allows a system to choose one DAS over another in many cases, but what if competing DASes have the same priority? In so-called *fixed priority* default theories (Horty 2012), there is no other option: an arbitrary decision must be made. Instead, a flexible reasoner might try to compare things such as the size of the preconditions met, or some measure of "activeness" to simulate the human bias known as the availability heuristic.

4 Making it Happen: Flexible Reasoning with MATR

One thing is clear about reasoning over DASes in $\mathcal{CEC}$: simply plugging axioms into a standard theorem prover will likely be slow. My initial explorations with this in the SPASS-based prover TALOS (a prover designed for use with $\mathcal{DCEC}^*$ (Licato 2015)) timed out without completing its proofs virtually every time. It seems that to perform, e.g., the confounder searching and plan elaboration as described in Section 3.4, what is needed is a very directed reasoner, one that knows in a sense what it's trying to prove (e.g. that it's trying to specifically find things to expand the plan, and how to find confounders).

Perhaps the clearest example of the need for flexibility comes from the evaluation of the `consistent` function,

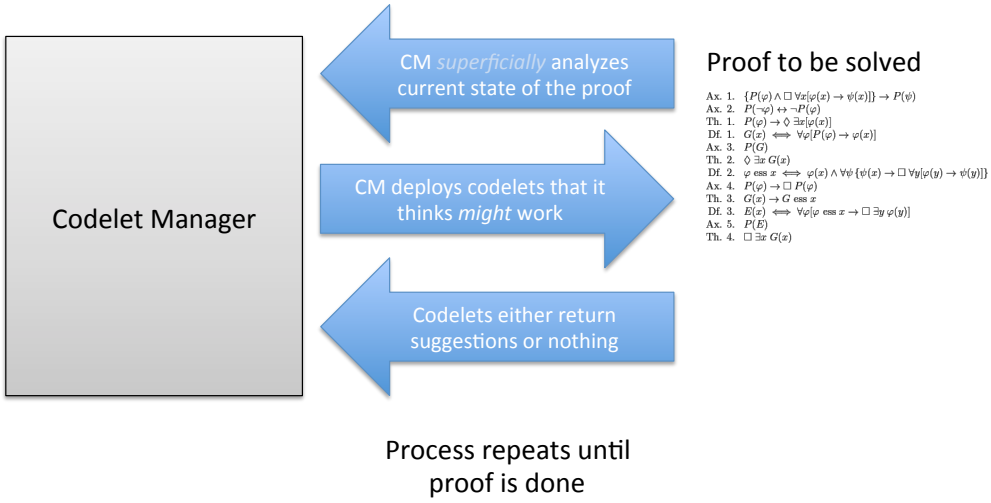

Figure 3: The Codelet Manager in MATR Deploys and Coordinates Codelets

which is very implementation-specific and may need to be configurable based on the needs of some particular domain.

Only one such prover, to my knowledge, has the flexibility to reason in such a way with minimal reprogramming effort, performing fast searches for confounders while preserving its ability to be a general-purpose reasoner. That reasoner is MATR (Machina Arachne Tree-based Reasoner). MATR is currently being developed by myself and researchers at the Rensselaer AI and Reasoning (RAIR) lab, having recently finished an early alpha version.

I believe a general-purpose, argument-theoretic natural reasoner like MATR is the only way forward for $\mathcal{CEC}$ and the other cognitive calculi, given that augmentations such as the DASes introduced in this paper will likely become more numerous, and each such augmentation may require unique methods of reasoning and tailorable search strategies to make their inclusion in real-time systems feasible. MATR allows such reasoning methods and search strategies to run in parallel by outsourcing the bulk of its heavy lifting to *codelets*. Combinations of codelets can be deployed depending on the high-level needs of MATR's end user (e.g. selecting codelets for general-purpose $\mathcal{CEC}$ reasoning versus finding proofs in propositional calculus) or the dynamically changing needs of a specific reasoning session (e.g. certain codelets may be better to run at the beginning, middle, or end of a proof).

The *codelet manager* component controls the deployment and coordination of codelets once a reasoning session has begun (Figure 3). During a reasoning session, the codelet manager will constantly scan the current state of the inference space (these scans will often be superficial for speed reasons) and use adaptive metrics to determine which codelets are most appropriate to start. The codelet manager's suggestions may not be optimal, but the overall strategy is one of massive parallelization—many strategies are deployed simultaneously and their results synthesized, and this process is repeated over many iterations.

The granularity of codelets is variable: an API will be available to develop codelets in Java, along with example codelets that are very low-level, corresponding to individual inference rules so that the resulting proof found by MATR is one resembling natural deduction, or high-level, such as a codelet which calls another theorem prover entirely. Furthermore, codelets are not limited to deduction; initial explorations into developing codelets for inductive and analogico-deductive (Licato, Bringsjord, and Hummel 2012) reasoning are already underway.

5 Conclusion and Future Work

There are three primary contributions of this paper: First, this is to my knowledge the first instance of action schemas, and reasoning over them, in one of the cognitive calculi. Second, this is the first time default reasoning has been attempted in the cognitive calculi, and finally, the discussion of how they can be used, particularly with special functions such as `consistent`, will be used as a starting point for future work. I introduced MATR as an example of a reasoner designed to handle the flexibility of reasoning that DASes will require.

The approach to DASes in $\mathcal{CEC}$, presented for the first time in this paper, will need some significant refinement. One potential problem may arise from the implicit assumption that if an agent believes some action schema, then the relevant background theory against which that DAS must be evaluated for consistency may not consist entirely of beliefs of the agent. More work will have to be done in this area to understand all the implications of this approach.

Finally, it should be noted that default reasoning has its share of problems: it fails some tests for defeasible reasoning such as cautious monotony and distribution (Koons 2014). Default reasoning is also not the only way to explain the types of reasoning discussed in this paper (see for example (Pollock 2000)). The limitations of default reasoning will

need to be explored in future work and compared to other reasoning types. It remains to be seen how well $\mathcal{CEC}$ (and the other cognitive calculi) can accommodate realistic non-monotonic reasoning in general.

References

Arkoudas, K., and Bringsjord, S. 2009. Propositional Attitudes and Causation. *International Journal of Software and Informatics* 3(1):47–65.

Bringsjord, S., and Ferrucci, D. 1999. *Artificial Intelligence and Literary Creativity: Inside the Mind of Brutus, A Storytelling Machine*. Psychology Press.

Bringsjord, S.; Govindarajulu, N. S.; Ellis, S.; McCarty, E.; and Licato, J. 2014. Nuclear Deterrence and the Logic of Deliberative Mindreading. *Cognitive Systems Research* 28:20–43.

Bringsjord, S.; Govindarajulu, N. S.; Licato, J.; Sen, A.; Johnson, J.; Bringsjord, A.; and Taylor, J. 2015. On Logicist Agent-Based Economics. In *Proceedings of Artificial Economics 2015 (AE 2015)*. Porto, Portugal: University of Porto.

Castelfranchi, C. 2000. Artificial Liars: Why Computers Will (Necessarily) Deceive Us and Each Other. *Ethics and Information Technology* 2:113–119.

Chisholm, R. M. 1963. Contrary-to-Duty Imperatives and Deontic Logic. *Analysis* 24:33–36.

Clark, M., and Atkinson, D. J. 2013. (Is There) A Future for Lying Machines? In *Proceedings of the 2013 Deception and Counter-Deception Symposium*.

Cohen, G. 2002. Deeper Into Bullshit. In Buss, S., and Overton, L., eds., *Contours of Agency: Essays on Themes from Harry Frankfurt*. Cambridge, MA: MIT Press. 321–339.

Horty, J. F. 2012. *Reasons as Defaults*. Oxford University Press.

Jaśkowski, S. 1934. On the Rules of Suppositions in Formal Logic. *Studia Logica* 1:5–32.

Koons, R. 2014. Defeasible Reasoning. In Zalta, E. N., ed., *The Stanford Encyclopedia of Philosophy*. Stanford University, spring 2014 edition.

Kowalski, R., and Sergot, M. 1986. A Logic-based Calculus of Events. *New Generation Computing* 4(1):67–94.

Licato, J.; Bringsjord, S.; and Hummel, J. E. 2012. Exploring the Role of Analogico-Deductive Reasoning in the Balance-Beam Task. In *Rethinking Cognitive Development: Proceedings of the 42nd Annual Meeting of the Jean Piaget Society*.

Licato, J.; Sun, R.; and Bringsjord, S. 2014. Using Meta-Cognition for Regulating Explanatory Quality Through a Cognitive Architecture. In *Proceedings of the 2nd International Workshop on Artificial Intelligence and Cognition*.

Licato, J. 2015. Talos prover page. http://rair.cogsci.rpi.edu/projects/automated-reasoners/talos/.

McNamara, P. 2014. Deontic Logic. In Zalta, E. N., ed., *The Stanford Encyclopedia of Philosophy*. Stanford University, winter 2014 edition.

Pollock, J. L. 2000. Rational Cognition in OSCAR. *Lecture Notes in Computer Science* 1757:71–90.

Reiter, R. 1980. A Logic for Default Reasoning. *Artificial Intelligence* 13:81–132.

Russell, S. J., and Norvig, P. 2010. *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, NJ: Prentice Hall, 3 edition.

Sharkey, A., and Sharkey, N. 2011. Anthropomorphism and Deception in Robot Care and Companionship. *IEEE Robots and Automation Magazine* 18(1):32–38.

Wagner, A. R., and Arkin, R. C. 2009. Robot Deception: Recognizing when a Robot Should Deceive. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA-09)*.