

Action-Centered Information Retrieval

MARCELLO BALDUCCINI

Saint Joseph's University
marcello.balduccini@gmail.com

EMILY C. LEBLANC

Drexel University
ec138@drexel.edu

submitted 1 January 2003; revised 1 January 2003; accepted 1 January 2003

Abstract

Information Retrieval (IR) aims at retrieving documents that are most relevant to a query provided by a user. Traditional techniques rely mostly on syntactic methods. In some cases, however, links at a deeper semantic level must be considered. In this paper, we explore a type of IR task in which documents describe sequences of events, and queries are about the state of the world after such events. In this context, successfully matching documents and query requires considering the events' possibly implicit, uncertain effects and side-effects. We begin by analyzing the problem, then propose an action language based formalization, and finally automate the corresponding IR task using Answer Set Programming. *Under consideration in Theory and Practice of Logic Programming (TPLP).*

KEYWORDS: Reasoning about Actions and Change, Answer Set Programming, Information Retrieval

1 Introduction

Information Retrieval (IR) (Korfhage, 1997) aims at identifying, among a set of available information sources, those that are most relevant to a query provided by the user. IR is arguably a staple of every day life – we consult Wikipedia for general reference, doctors search private databases for patient information, and researchers use public databases to find scientific publications. IR is also at the core of numerous commercial activities such as searching for news about business partners or competitors.

Most IR systems base the relevance of a source on a syntactic measurement of the overlap of terms between query and source (Manning et al., 2008). Even advanced techniques still focus on syntactic matching, and include temporal IR (Campos, 2015), query expansion (Carpineto and Ramano, 2012), and graph-based term weighting (Blanco and Lioma, 2012).

Recent research (Glavas and Snajder, 2014) has demonstrated that traditional IR yields low accuracy when applied to documents centered on events, such as police reports, medical records, and breaking news. As one can imagine, documents centered on events occur in large quantities and often contain very valuable information. Consider an example related to healthcare: a radiologist might be looking for information on whether a patient was ever bedridden. This type of information is rarely stated explicitly in patient documents; rather, the radiologist is more likely to have access to documents reporting, for instance, that the patient suffered a multiple fracture at his left leg. Such a document is indeed relevant to the radiologist's request, since the patient

was likely bedridden as a result of the injury. However, determining its relevance requires linking the event of suffering a leg injury to the resulting state of the patient, which is beyond the reach of IR techniques based on syntactic measurements. Similar considerations can be made in the context of cybersecurity/cyberanalytics. Consider the case of a user asking whether a computer was left without network connectivity during a certain time frame. A system log stating that the router, to which the computer is connected, was restarted during that period of time, is indeed relevant to the user’s query. However, identifying its relevance requires the ability to consider that a restart causes the router to transition to a state in which connected devices are without network connectivity. Again, this capability is clearly beyond the reach of traditional IR. In fact, in practice this use case may be even more complicated if the computer is connected to the router through other devices, in which case identifying a match requires reasoning about how the loss of connectivity propagates recursively to any device connected to the router.

Glavas and Snajder (2014) proposed a new approach, called event-centered IR, which succeeded in increasing match accuracy by means of some level of semantic analysis. However, their approach was limited to matching events mentioned in both queries and sources. This is insufficient to handle the above examples, where one needs to link events and information about the *state of the world* before, during, and after the events.

In this paper, we advance this line of research by considering the case in which the goal is to match sources containing sequences of events with queries that are about the state of the world after those events. The examples provided above fall in this category, as well as, generally speaking, all cases in which the sources describe the history of a domain (e.g., historical documents about property sales, police reports, computer event logs) and a user is interested in those sources from which the state of the world at a certain point in time can be reconstructed (e.g., “was the firewall active when the attack happened?”).

Our approach aims to identify reasonable matches even when a definitive answer cannot be immediately found in the sources, events have complex/hidden effects, and information is incomplete. We call the corresponding reasoning task *Action-Centered IR*.

The aim of the present paper is to provide an accurate definition of the problem, of the corresponding reasoning tasks, and of algorithms for automating the process. We begin the paper by analyzing the problem and, appealing to commonsense and intuition, identify reasonable outcomes of the task as a human reader might carry it out. For illustration purposes, we start with a simple problem, which we progressively elaborate. An exhaustive evaluation of our approach over realistic examples is beyond the scope of the paper: as discussed by Glavas and Snajder (2014), the existing benchmarks for IR tasks are not suitable for the evaluation of semantic-level matching approaches and the development of suitable datasets is a research project in its own merit, which we will tackle and document separately. However, we conducted a preliminary investigation of the scalability of our approach, whose results are discussed later in the paper.

It should be noted that, throughout the paper, we assume that passages in natural language have already been translated into a suitable logic form, as the natural language task is orthogonal to the problem addressed here. Specifically, we assume that query and sources have already been translated to a temporally-tagged logical representation, e.g., by using techniques from (Nguyen et al., 2015; LeBlanc and Balduccini, 2016; Lierler et al., 2017). We also assume the availability of suitable knowledge repositories (Matuszek et al., 2006; Suchanek et al., 2008; Inclezan, 2016). Additionally, while our work is somewhat related to research on temporal relations (e.g., Allen’s interval calculus), the two differ because we focus on reasoning about events and their effects, rather than relations between events.

The main contributions of this paper are (a) the exploration of a novel, non-trivial variant of IR in which sources include sequences of events, and queries are about the state of the world after such events; (b) the extension of techniques for representing dynamic domains to increase the flexibility of reasoning in the presence of uncertainty; (c) a formalization of the IR task based on action languages; (d) an automated IR procedure based on Answer Set Programming (ASP).

In the next section, we cover the analysis of the problem. Next, we present a formalization of the relevant knowledge and of the reasoning tasks underlying Action-Centered IR. In the following section, we illustrate an ASP-based procedure for automating the reasoning processes, and demonstrate it on the toy scenarios. Then, we present an experimental evaluation of our approach. Finally, we discuss related work and draw conclusions.

2 Problem Analysis

In this section, we proceed by example to an analysis of the problem of Action-Centered IR and discuss, in commonsensical terms, the underlying reasoning tasks. Let us start with the following:

Example 1

The user's query, q , is "Is John married?" Available information sources are:

S_1 : "John went on his first date with Mary."

S_2 : "John read a book."

We want to determine which source is most relevant to q .

The query refers to the current state of the world, which with some approximation we can identify with the final state of the world in the sources. The sources describe events that occurred over time. Neither source mentions being married, making syntactic-based methods unfit for the task. However, from an intuitive perspective, S_1 is more relevant to q than S_2 . In fact, S_1 , together with commonsense knowledge that married people typically do not go on first dates, provides a strong indication that John is not married. S_2 , on the other hand, provides no relevant information.

In this simple example, one can not only identify S_1 as the most relevant source, but also look for an accurate answer to the question. The simplicity of the example blurs the line between IR and question answering. In general, however, providing an accurate answer requires a substantial amount of reasoning to be carried out once a relevant source has been identified, as well as deep understanding of the content of the source and a large amount of world knowledge – something that is still challenging for state-of-the-art approaches. Thus, in this paper, *we assume that a reader with human-level intelligence will later find accurate answers by studying the sources identified as relevant by our approach. We focus on defining techniques that provide the reader with a ranking of the sources based on our expectation that answers may be found in them.*

The previous example allows us to establish a first, high-level characterization of the task we aim to study, as one in which we are given a query q and a collection of sources S_1, \dots, S_n , and are asked to produce scores s_1, \dots, s_n ranking each source based on its relevance to the problem of finding an answer to q . If we adopt the convention that 0 is the best possible score and ∞ the worst, then it is conceivable that, in Example 1, S_1 should be assigned a score of 0 and S_2 a score of ∞ to indicate complete irrelevance.

As in traditional IR, the sources are ranked based on their respective score. We expect that, in the long-term, both syntactic and semantic aspects will have to be taken into account to determine scores for the documents. Thus, below, we use the term "semantic score" when we refer to the score assigned to documents by the techniques we study. It is worth stressing the difference

between the task at hand and question answering, where the goal is to produce a definitive answer. At the end of the process we consider here, the answer to q may still be unknown, but there will be reason to believe that careful study of the sources identified as relevant will lead to an answer.

Next, we consider a number of examples and corresponding expectations. Based on the examples, later we propose a formalization of the reasoning processes. Example 1 showed that the event of going on a first date may lead us to infer that John is not married. But how can one reach such conclusion? One option is to reason by cases, and consider two possible views of the world: one in which John is married at the beginning of the story, and one in which he is not. Commonsense tells us that the action¹ of going on a first date is not executable when married. Hence, the view in which John is initially married is inconsistent with the source. So, we conclude that John must not have been married in the initial state. Given further knowledge that one does not get married on a first date, one can infer that John remains not married after the date. Thus, the source provides evidence that a reader can use to answer the query.

From a technical perspective, the example highlights the importance of being able to reason by cases, to reason about the executability of actions, and to propagate the truth of properties of interest over the duration of the story. Note, however, that reasoning by cases is sometimes misleading. Consider \mathcal{S}_2 from Example 1: reasoning by cases leads to the same two possible initial states. Since reading does not affect married status, there are two ending states for the story. This might be taken as an indication that the source provides *some* useful evidence for a reader, but it is clear intuitively that \mathcal{S}_2 is, in fact, irrelevant. Next, let us consider if, and how, the previous query should match a more complex document. For the sake of this example, let us assume the existence of a fictitious country C, whose laws allow plural marriage.

Example 2

q : *Is John married?*

\mathcal{S} : *John, who lives in country C, just went on his first date with Mary.*

In this case, \mathcal{S} does not provide useful information towards answering q . John is from C, where plural marriage is allowed, and knowledge about plural marriage yields that being married does not preclude a married person from going on a first date. The example also demonstrates the importance of reasoning about *default statements* (statements that are *normally* true) and their exceptions. The fact that married people typically do not go on first dates is an instance of a default statement, and an inhabitant of C constitutes an exception to it. Similarly to \mathcal{S}_2 from the previous example, reasoning by cases may be somewhat misleading, as it may suggest that the source provides some evidence useful to answering the question. Rather than reasoning by cases, it appears to be more appropriate to state that whether John is initially married is *unknown*. The lack of knowledge is propagated to the final state, given that going on a date has no effect on it in the present context. The source is thus irrelevant and should receive a semantic score of ∞ . Note the striking difference in scores between \mathcal{S}_1 from the previous example and the current source: it appears that in some cases reasoning by cases is useful, while in others reasoning explicitly about lack of knowledge is more appropriate. In the next section, we provide a characterization of reasoning matching this intuition. Next, we investigate the role of the effects of actions.

Example 3

¹ From now on, we will use action and event as synonyms.

q: Is John married?

S: John, who lives in country C, recently went on his first date with Mary. A week later, they tied the knot in Las Vegas.

Obviously, a first indication of relevance can be obtained with shallow reasoning and syntactic matching: “tying the knot” is a synonym of “getting married,” and “getting married” and “being married” share enough similarities to make a match likely. However, we are interested in more sophisticated reasoning. In the initial state, John may or may not be married due to his country’s laws. Similarly to Example 1, John’s married status persists in the state following the first date. Tying the knot, however, has the effect of making John married in the resulting state. Hence, \mathcal{S} is indeed relevant to q . Intuitively, its semantic score should be equal to that of \mathcal{S}_1 from Example 1. This demonstrates the importance of keeping track of the changes in the truth of the relevant properties over time. The next example takes this argument one step further.

Example 4

q: Is John married?

S: John recently went on his first date with Mary. A week later, they tied the knot in Las Vegas. A month later, they filed for divorce.

Here, we assume that filing for divorce does not immediately cause the spouses to be divorced. For simplicity, we adopt a view in which filing has a non-deterministic effect: in the resulting state, it is equally likely for the spouses to be married or not. The relevance of \mathcal{S} to q is not as straightforward as in previous cases. It is true that, at the end of the story, it is unknown whether John is married. On the other hand, the story still provides some information pertaining to John’s married status – certainly, more than source \mathcal{S}_2 (“John read a book”) from Example 1 or the source from Example 2 (“John, who lives in country C, just went on his first date with Mary.”).

One way to make a distinction between the two cases is to consider that, if \mathcal{S} from Example 4 is provided to a reader, and the reader manages to determine if the filing action succeeded (e.g., by gathering additional evidence), \mathcal{S} will immediately allow the reader to answer q . Differently from the previous examples, knowing that filing occurred is *essential* to allowing a reader to answer the question. Hence, while \mathcal{S} is not as relevant to q as other sources we have considered, it is still somewhat relevant. This will have to be reflected in the score assigned to the source, which should be higher than the 0 assigned to \mathcal{S}_1 , but obviously smaller than ∞ because the source is indeed relevant. Next, we propose a formalization that captures the behaviors described.

3 Formalization of the Action-Centered IR

One may note that carrying out the reasoning discussed above requires considering how actions may affect the state of the world in possibly indirect and intricate ways. For this reason, our formalization of the reasoning task at the core of Action-Centered IR leverages techniques from the research on reasoning about actions and change, and specifically action language \mathcal{AL} (Baral and Gelfond, 2000), approximated representations (Morales et al., 2007) and evidence-based reasoning (Balduccini and Gelfond, 2003). These techniques rely on a graph-based representation of the evolution of the state of the world over time in response to the occurrence of actions. We adopt and expand this approach. Specifically, similarly to the approach by Morales et al. (2007), our formalization enables reasoning explicitly about lack of knowledge. Differently from it, however, we allow a reasoner to reason by cases whenever needed. This is applied to knowledge about both

initial state and effects of actions. Our approach also leverages evidence-based reasoning to rule out some of the cases considered. Finally, we adopt \mathcal{AL} as the underlying formalism, but expand it for an explicit characterization of non-deterministic effects and we allow hypothesizing about exceptional/atypical circumstances, eventually linking them to the relevance of sources. Differently from \mathcal{AL} , our language is defined so that, in the presence of actions with non-deterministic effects, it is possible to reason by cases as well as by explicitly characterizing lack of knowledge. The syntax of the language, which we call \mathcal{AL}_{IR} , is described next, followed by its semantics.

Let \mathcal{F} be a set of symbols for *fluents* and \mathcal{E} be a set of symbols for *elementary actions*. Fluents are boolean properties of the domain, whose truth value may change over time. An *action* is a set of elementary actions. With slight abuse of notation, we denote a singleton action by its element.

A *fluent literal* is a fluent f or its negation $\neg f$. The complement of f (written \bar{f}) is $\neg f$, and vice-versa. The set of literals formed from \mathcal{F} is denoted by *Lit*. An *extended (fluent) literal* is either a fluent literal or an expression of the form $u(f)$, intuitively meaning that it is unknown whether fluent f is true or false. An expression $u(f)$ is called a *proper extended literal*.

A *signature* is a tuple $\Psi = \langle \mathcal{F}, \mathcal{E} \rangle$, whose components are defined above. Given a signature, the *laws* of \mathcal{AL}_{IR} are statements of the form:

$$e \text{ causes } \lambda \text{ if } l_1, l_2, \dots, l_n \quad (1)$$

$$l_0 \text{ if } l_1, \dots, l_n \quad (2)$$

$$e \text{ impossible if } l_1, \dots, l_n \quad (3)$$

where λ is an extended literal, l_1, \dots, l_n are fluent literals, and e is an elementary action². Law (1) is called *dynamic (causal) law*. If λ is a fluent literal, the law intuitively says that, if action e is executed in a state in which l_1, \dots, l_n hold, then λ will hold in the next state. If λ is a proper extended literal $u(f)$, the law intuitively states that the action affects the truth of f non-deterministically. λ is called the *consequence of the law*. The action of filing for divorce from Example 4 might be formalized with a dynamic law that has $u(\text{married})$ as its consequence. Law (2) is called *state constraint* and says that, in any state in which l_1, \dots, l_n hold, l_0 also holds. As in \mathcal{AL} , state constraints allow for an elegant and concise representation of the indirect effects of actions, increasing the expressive power of the language significantly. Law (3) is called *executability condition* and intuitively says that e cannot be executed if l_1, \dots, l_n hold. A set of laws of \mathcal{AL}_{IR} is called *action description*.³ The semantics of \mathcal{AL}_{IR} maps action descriptions to transition diagrams, as described next.

A set S of extended literals is *closed under a state constraint* (2) if $\{l_1, \dots, l_n\} \not\subseteq S$ or $l_0 \in S$. A set S of extended literals is *consistent* if, for every $f \in \mathcal{F}$, at most one of $\{f, \neg f, u(f)\}$ is in S . It is *complete* if at least one of $\{f, \neg f, u(f)\}$ is in S . A *state* of an action description AD is a complete and consistent set of extended literals closed under the state constraints of AD .

Given an action a and a state σ , the set of (*direct*) *effects of a in σ* , denoted by $E(a, \sigma)$, is the set that contains an extended literal λ for every dynamic law (1) s.t. $\{l_1, \dots, l_n\} \subseteq \sigma$ and $e \in a$.

Consider $A = \{A_1, A_2, \dots, A_k\}$ where every A_i is a set of extended literals. Let B be a set of extended literals. We define $A \boxtimes B = \{A_i \cup \{b\} \mid A_i \in A, b \in B\}$. For instance:

$$\bullet \{\{p\}, \{q\}\} \boxtimes \{r, \neg r\} = \{\{p, r\}, \{p, \neg r\}, \{q, r\}, \{q, \neg r\}\}$$

² We focus on elementary actions for simplicity of presentation. Expanding the laws to allow non-elementary actions is not difficult.

³ Technically speaking, a set of laws of \mathcal{AL}_{IR} should always be accompanied by a specification of a signature. For simplicity, we omit the signature whenever possible and infer it from the statements.

- $\{\{p, q\}\} \bowtie \{r, \neg r\} \bowtie \{s, \neg s\} = \{\{p, q, r, s\}, \{p, q, r, \neg s\}, \{p, q, \neg r, s\}, \{p, q, \neg r, \neg s\}\}$

Definition 1

Let a be an action and σ be a state. The *expansion of $E(a, \sigma)$* is:

$$\mathbb{E}(a, \sigma) = \{E(a, \sigma) \cap Lit\} \bowtie \{f_1, \neg f_1, u(f_1)\} \bowtie \dots \bowtie \{f_k, \neg f_k, u(f_k)\}$$

where $\{f_1, \dots, f_k\}$ is the set of fluents such that $u(f_i) \in E(a, \sigma)$ for every $1 \leq i \leq k$.

(For an illustration of the notion of expansion, refer to Example 5 below.) Given a set S of extended literals and a set Z of state constraints, the *set, $Cn_Z(S)$, of consequences of S under Z* is the smallest set of extended literals that contains S and is closed under Z . Finally, an action a is *executable* in a state σ if there is no executability condition (3) such that $\{l_1, \dots, l_n\} \subseteq \sigma$ and $e \in a$.

The semantics of an action description AD is defined by its *transition diagram* $\tau(AD)$, i.e., a directed graph $\langle N, E \rangle$ such that:

- N is the collection of all states of AD , and
- E is the set of all triples $\langle \sigma, a, \sigma' \rangle$ where σ, σ' are states, a is an action executable in σ , and σ' satisfies the *expanded successor-state equation*:

$$\sigma' = Cn_Z(W \cup (\sigma \cap \sigma')) \text{ for some } W \in \mathbb{E}(a, \sigma). \quad (4)$$

As before, Z is the set of all state constraints of AD . The argument of Cn_Z in (4) is the union of (i) the set of direct effects $E(e, \sigma)$ for each $e \in a$ with (ii) the set $\sigma \cap \sigma'$ of the facts “preserved by inertia”. The application of Cn_Z adds the “indirect effects” to this union. Triple $\langle \sigma, a, \sigma' \rangle$ is called a *transition of $\tau(AD)$* and σ' is a *successor state of σ* (under a). A *path* in a transition diagram $\mathcal{T}(AD)$ is a sequence $\pi = \langle \sigma_0, a_0, \sigma_1, a_1, \sigma_2, \dots, \sigma_n \rangle$ in which every triple $\langle \sigma_i, a_i, \sigma_{i+1} \rangle$ satisfies the expanded successor state equation. We denote the initial state of a path π by π_{σ_0} .

Example 5

Consider an action description $\{e_1 \text{ causes } f_1; e_1 \text{ causes } u(f_2); f_3 \text{ if } f_1\}$, a state $\sigma_0 = \{\neg f_1, \neg f_2, \neg f_3\}$, and action $a_0 = \{e_1\}$. One can check that $E(a_0, \sigma_0) = \{f_1, u(f_2)\}$. Note the lack of knowledge about f_2 , due to the non-deterministic nature of e_1 . Our definition of transition incorporates the idea that, in the presence of uncertainty about the effects of an action, one may sometimes model that uncertainty explicitly by means of $u(f)$ literals, and sometimes reason by cases by “replacing” $u(f)$ by f and $\neg f$. The set $\mathbb{E}(a_0, \sigma_0)$ captures this intuition, yielding the three possible options $\{f_1, u(f_2)\}$, $\{f_1, f_2\}$, and $\{f_1, \neg f_2\}$. Through (4), each option leads to a different successor state, $\{f_1, u(f_2), f_3\}$, $\{f_1, f_2, f_3\}$, and $\{f_1, \neg f_2, f_3\}$, obtained by taking into account the consequences of state constraints. Figure C 1 (see Appendix C) illustrates the corresponding transitions.

Intuitively, the first state from Example 5 is the most “economical,” in that it is obtained with the least amount of assumptions, while the other two are less “economical.” To keep track of where reasoning by cases is applied, we introduce the following definition.

Definition 2

The *branching-set* of a transition $\langle \sigma, a, \sigma' \rangle$ is:

$$\beta(\langle \sigma, a, \sigma' \rangle) = \{f \mid u(f) \in E(a, \sigma) \text{ and } u(f) \notin \sigma'\}$$

For Example 5, the branching-set for the first successor state considered is \emptyset , while for the other two it is $\{f_2\}$, indicating reasoning by cases over f_2 .

We call an action description AD *non-deterministic* when multiple successor states exist for a given σ and a . Furthermore, AD has *emergent non-deterministic behavior* if, for some a and σ , there exist multiple σ' such that the following equation (Baral and Gelfond, 2000) is satisfied:

$$\sigma' = Cn_Z(E(a, \sigma) \cup (\sigma \cap \sigma')). \quad (5)$$

In this paper, we focus on action descriptions without emergent non-deterministic behavior.⁴

Next, we turn our attention to the use of transition diagrams to reason about sequences of actions and to determine the relevance of available sources.

4 Reasoning about Relevance of Sources

In order to enable reasoning about the relevance of sources, we begin by formalizing the notions of source and query. A source is a tuple $\langle \Psi, \mathcal{D}, AD, I, \aleph \rangle$, where Ψ is a signature, \mathcal{D} is a (possibly empty) subset of \mathcal{F} (called the set of *default fluents*), AD is an action description, I is a consistent set of fluent literals (intuitively capturing the available information about the initial state of the source), and $\aleph = \langle a_0, a_1, \dots, a_k \rangle$ is a sequence of actions (which occur in the source). A query q is a fluent. Intuitively, default fluents are fluents that should be assumed to be false by default at the beginning of a source.

In our approach, a *qualified action sequence* is a tuple $s = \langle a_0/q_0, a_1/q_1, \dots, a_k/q_k \rangle$ where each a_i is an action and each q_i is a set of fluents, called *qualifier*. Intuitively, a qualifier specifies to which effects of the corresponding action one should apply reasoning by cases. In reference to Example 5, the expression $e_1/\{f_2\}$ intuitively means that the reasoner should consider the transitions in which f_2 and $\neg f_2$ hold in the successor state, while $e_1/\{\}$ indicates that only the transition resulting in $u(f_2)$ should be considered. The *length* of s is $k+1$. The *branching degree* of s is $\Delta(s) = |q_0| + |q_1| + \dots + |q_k|$. Given a sequence of actions $\aleph = \langle a_0, a_1, \dots, a_k \rangle$, we say that $s = \langle a_0/q_0, a_1/q_1, \dots, a_k/q_k \rangle$ *extends* \aleph for every possible choice of qualifiers. $\aleph^?$ denotes the extension where all qualifiers are $\{\}$ and \aleph^\times denotes the extension where all are \mathcal{F} . Let σ be a state and s be a qualified action sequence. A path $\pi = \langle \sigma_0, a_0, \sigma_1, \dots, a_k, \sigma_{k+1} \rangle$ is a *model* of $[\sigma, s]$ if (a) $\sigma_0 = \sigma$, and (b) $\beta(\langle \sigma_i, a_i, \sigma_{i+1} \rangle) = q_i$. Given a set Σ of states and a qualified action sequence s , a path π is a *model* of $[\Sigma, s]$ if π is a model of $[\sigma, s]$ for some $\sigma \in \Sigma$. Consider an action description $\{a_1 \text{ causes } \neg g \text{ if } g; a_2 \text{ causes } u(f) \text{ if } \neg g\}$. Let σ be $\{\neg f, g\}$. One can check that $s_1 = [\sigma, \langle a_1/\emptyset, a_2/\emptyset \rangle]$ has a unique model, $\langle \{\neg f, g\}, a_1, \{\neg f, \neg g\}, a_2, \{u(f), \neg g\} \rangle$. On the other hand, $s_2 = [\sigma, \langle a_1/\emptyset, a_2/\{f\} \rangle]$ has two models, $\langle \{\neg f, g\}, a_1, \{\neg f, \neg g\}, a_2, \{f, \neg g\} \rangle$ and $\langle \{\neg f, g\}, a_1, \{\neg f, \neg g\}, a_2, \{\neg f, \neg g\} \rangle$. Hence, $\Delta(s_1) = 0$ and $\Delta(s_2) = |\emptyset| + |\{f\}| = 1$.

Now we turn our attention to incomplete knowledge about the initial state. In our approach, the *default assumption* is to consider the default fluents false and to assume that $u(f)$ holds for every non-default fluent f . However, as highlighted by Section 2, commonsense sometimes leads one to explore cases beyond those of the default assumption – either considering that a default fluent might be true, or reasoning by cases over the truth of the other fluents.

This intuition is captured by the notion of *forcing of a fluent*. A fluent whose truth value is currently unknown is *forced* when a reasoner wants to consider for it cases other than those from

⁴ Action description $\{q \text{ if } \neg r, p; r \text{ if } \neg q, p; a \text{ causes } p\}$ has an emergent non-deterministic behavior.

the default assumption. More precisely, let I be a consistent set of extended literals and f be a fluent that should be forced. The *forcing of f in I* , written $I[f]$, is defined as follows:

$$I[f] = \begin{cases} \{I \cup \{f\}\} & \text{if } f \in \mathcal{D} \text{ and } \{\neg f, u(f)\} \cap I = \emptyset \\ \{I \cup \{f\}, I \cup \{\neg f\}\} & \text{if } f \notin \mathcal{D} \text{ and } \{f, \neg f, u(f)\} \cap I = \emptyset \\ \{I\} & \text{otherwise} \end{cases}$$

For an example of the notion of forcing, consider the following.

Example 6

Consider \mathcal{S}_1 from Example 1, “John went on his first date with Mary.” The source is encoded by tuple $\langle \Psi, \mathcal{D}, AD, I, \aleph \rangle$ where $\Psi = \langle \mathcal{F}, \mathcal{D}, \mathcal{E} \rangle$ and:

- \mathcal{F} consists of fluents: m – John is married⁵; ab – John is an exception w.r.t. going on first dates when married.
- $\mathcal{D} = \{ab\}$, i.e. by default, John is not an exception.
- \mathcal{E} consists of action d , i.e. going on a first date.
- Action description AD consists of law $\{\text{impossible } d \text{ if } m, \neg ab\}$, intuitively stating that a married person does not normally go on first dates.⁶
- The knowledge about the initial state of the source is captured by $I = \emptyset$.
- The sequence of actions is $\aleph = \langle d \rangle$, i.e. John went on a first date.

Finally, query q is given by fluent m . Because the story does not say whether m holds in the initial state, the forcing of m in I allows one to consider both cases explicitly. One can check that $I[m]$ is $\{\{m\}, \{\neg m\}\}$.

It is important to note that fluents that already occur in I are not affected by the forcing. The notion of forcing is extended to sets of fluents in a natural way. The *forcing of $\{f_1, \dots, f_m\}$ in I* is defined recursively as follows:

$$I[\{f_1, \dots, f_m\}] = \begin{cases} I[f_1] & \text{if } m = 1 \\ \{I'[f_m] \mid I' \in I[\{f_1, \dots, f_{m-1}\}]\} & \text{if } m > 1 \end{cases}$$

In the case of Example 6, one can check that $I[m, ab]$ is $\{\{m, ab\}, \{\neg m, ab\}\}$, intuitively meaning that both cases are considered for m and that default fluent ab is hypothesized to be true.

Once the fluents that deserve special treatment have been handled, the default assumption is applied to all remaining fluents whose truth value is still unknown. This process is captured by the notion of completion, defined next.

Definition 3

Let I be a consistent set of fluent literals and Z be the set of state constraints of an action description AD . The *completion of I* , denoted by $\gamma(I)$, is the consistent set of extended literals obtained as follows:

1. Let I' be obtained by expanding I with a fluent literal $\neg f$ for every default fluent f such that $f \notin I$.
2. If $Cn_Z(I')$ is consistent, then $\gamma(I)$ is the union of $Cn_Z(I')$ and a literal $u(f)$ for every f that does not occur in $Cn_Z(I')$. Otherwise, $\gamma(I)$ does not exist.

⁵ In practical cases, one will want to introduce variables to increase generality, e.g. $m(X)$ for X is married.

⁶ Note the use of default fluent ab to formalize the fact that action d is *normally* impossible if one is married.

For an example of a case in which $\gamma(I)$ does not exist, consider $I = \{p, q\}$, $\mathcal{D} = \emptyset$ and $AD = \{-q \text{ if } p\}$. Given that there are no default fluents, $I' = I$. The application of Cn_Z to I' results in $\{p, q, \neg q\}$, which is inconsistent. Hence, the completion of I does not exist.

When $\gamma(I)$ exists, it is not difficult to check that the following holds:

Proposition 1

For any consistent set of fluent literals I , $\gamma(I)$ is complete, consistent and includes I .

We can now combine forcing of a set of fluents F and completion of its outcomes as follows:

Definition 4

Let F be a set of fluents. The *completion of I w.r.t. F* is the set:

$$\gamma(I, F) = \{\gamma(I') \mid I' \in I[F] \text{ and } \gamma(I') \text{ exists}\}.$$

The *degree of $\gamma(I, F)$* , denoted by $\Delta(\gamma(I, F))$, is $|F|$.

The following example illustrates this concept.

Example 7

Continuing Example 6, let us find $\gamma(I, F)$ for $F = \{m\}$. Intuitively, this means that we would like to consider explicitly the possible options for the truth value of m , while applying the default assumption to all other fluents.

According to Definition 4, first we need to find the forcing of F in I . By definition of forcing, $I[F] = I[m]$. In Example 6, we found $I[m]$ to be $I[m] = \{\{m\}, \{\neg m\}\}$. Hence, $I[F] = \{\{m\}, \{\neg m\}\}$.

Next, we apply the default assumption to every $I' \in I[F]$ by finding $\gamma(I')$. From Definition 3 it follows that $\gamma(\{m\}) = \{m, \neg ab\}$ and $\gamma(\{\neg m\}) = \{\neg m, \neg ab\}$. Hence, $\gamma(I, F) = \{\{m, \neg ab\}, \{\neg m, \neg ab\}\}$. Intuitively, this corresponds to a situation in which a reasoner considering possible initial states for the scenario makes the default assumption about default fluent ab , but decides to reason by cases about m .

As demonstrated by Example 1, there are cases in which the truth of certain fluents in the initial state can be inferred indirectly from the source. By building on the previous definitions, we can now make this idea precise in the following.

Definition 5

Given a consistent set I of fluent literals and a sequence of actions \aleph , the *conservative expansion of I under \aleph* is:

$$\varepsilon(I, \aleph) = \bigcap_{I' \in I[\mathcal{F} \setminus \mathcal{D}]} \{I' \mid [\gamma(I'), \aleph^\times] \text{ has a model}\}$$

The intuition behind this definition is that the reasoner expands I by considering all possible cases for the non-default fluents from I . For each resulting expanded set I' , the reasoner checks if there exists a completion $\gamma(I')$ compatible with the actions in \aleph . The conservative expansion of I under \aleph consists of the extended literals shared by all expanded sets I' satisfying this test. To ensure that all possible contingencies are considered, the definition applies reasoning by cases to the effects of the actions in \aleph – hence the use of \aleph^\times .

The above definition yields a number of important properties.

Proposition 2

1. If $\gamma(I')$ does not exist for any element of $I[\mathcal{F} \setminus \mathcal{D}]$, then $\varepsilon(I, \aleph)$ does not exist.
2. If $\varepsilon(I, \aleph)$ exists, then $I \subseteq \varepsilon(I, \aleph)$.
3. When it exists, $\varepsilon(I, \aleph)$ is consistent but not necessarily complete.

Note that, if $\varepsilon(I, \aleph)$ does not exist, this intuitively indicates that there is some fundamental inconsistency in the story and the source should be considered irrelevant to any query. As we will see later, this is handled by assigning a semantic score of ∞ to the source.

Example 8

Let us calculate $\varepsilon(I, \aleph)$ for our running example. Recall that $\aleph = \langle d \rangle$. The first step consists in checking for models of $[\gamma(I'), \aleph^\times]$ where $I' \in I[\mathcal{F} \setminus \mathcal{D}]$. From Example 7, we know that $I[\mathcal{F} \setminus \mathcal{D}] = \{\{m\}, \{\neg m\}\}$ and that the completions of each component of the set are, respectively, $\{m, \neg ab\}$ and $\{\neg m, \neg ab\}$. We can now check for models. Clearly, $[\{m, \neg ab\}, \langle d \rangle]$ has no model, because d is not executable in $\{m, \neg ab\}$. On the other hand, $[\{\neg m, \neg ab\}, \langle d \rangle]$ has a model. The second step consists in calculating the intersection of all I' that satisfy the requirements. In this example, $\varepsilon(I, \aleph)$ is the intersection of the only set $\{\neg m\}$, resulting in $\varepsilon(I, \aleph) = \{\neg m\}$. That is, we have inferred that John is not married in the initial state, which is aligned with the conclusion reached in Example 1.

We are now ready to introduce the notion of entailment and to use it to determine whether there is a match between \mathbf{q} and \mathcal{S} . A path $\pi = \langle \sigma_0, a_0, \sigma_1, \dots, a_{k-1}, \sigma_k \rangle$ entails a fluent literal l (written $\pi \models l$) if $l \in \sigma_k$. Given a fluent f , we say that π entails $\pm f$ (written $\pi \models \pm f$) if $\pi \models f$ or $\pi \models \neg f$. The main notion of this section is given next.

Definition 6

Given a source $\mathcal{S} = \langle \Psi, AD, I, \aleph \rangle$ and a query \mathbf{q} , we say that \mathcal{S} is a match for \mathbf{q} if there exist a set F of fluents from Ψ and a qualified action sequence s extending \aleph such that:

- c1** $\pi \models \pm \mathbf{q}$ for some model π of $[\gamma(\varepsilon(I, \aleph), F), s]$, and
- c2** for every model π' of $[\gamma(\pi_{\sigma_0} \setminus \varepsilon(I, \aleph), \emptyset), \langle \rangle]$, one of the following holds:
 - (a) $\pi' \not\models \pm \mathbf{q}$, or
 - (b) $\pi' \models \neg \mathbf{q}$ and $\pi \models \mathbf{q}$, or
 - (c) $\pi' \models \mathbf{q}$ and $\pi \models \neg \mathbf{q}$.

Condition **(c1)** checks whether the \mathcal{S} is relevant to \mathbf{q} , that is, if it has any bearing on the truth value of \mathbf{q} . To do so, the reasoner is allowed to reason by cases about an arbitrary set of non-default fluents and to assume that some default fluents behave exceptionally. This choice of fluents is embodied by set F and by its role in **(c1)**. The reasoner is also allowed to reason by cases about the effects of an arbitrary set of actions from \aleph , as outlined by the freedom in selecting its extension s . Note that the key criterion that a path needs to satisfy in **(c1)** is the entailment of $\pm \mathbf{q}$.

Condition **(c2)** ensures that the assumptions made by **(c1)** leading to the selection of path π , are not directly and solely responsible for the entailment of $\pm \mathbf{q}$. (Refer to Example 9 for an illustration of the application and interplay of **(c1)** and **(c2)**.) To achieve this, given path π , condition **(c2)** identifies the fluents of π_{σ_0} that are in I and those whose truth value was inferred according to Definition 5. The default assumption is applied to all of those fluents. Next, the truth value of \mathbf{q} is checked in the state obtained in this way. If the truth value of \mathbf{q} in this state coincides with the truth value of \mathbf{q} in π_{σ_0} , this indicates that the entailment of $\pm \mathbf{q}$ in **(c1)** was due solely to

the assumptions made and was independent from the initial state information and actions from \mathcal{S} . Therefore, the current model π should be discarded, as insufficient evidence that \mathcal{S} is a match for \mathbf{q} . \mathcal{S} is considered to be a match for \mathbf{q} only when a path satisfying (c1) is found and when is shown that the information from \mathcal{S} is critical in entailing $\pm\mathbf{q}$ in π_{σ_0} .

In Definition 6, F and s can be viewed as an indication of the “strength” of the match. If a match can be found for $F = \emptyset$ and $s = \mathbb{N}^?$, it means that all that is needed to determine that \mathcal{S} is a match for \mathbf{q} is the information from \mathcal{S} . This makes for a “strong match”. Instead, if a match is found only for other values of F and s , it means that the match depends on additional assumptions, such as assuming that a default fluent is unexpectedly true or that a non-default fluent has a specific truth value. This makes for a “weaker” match, since such assumptions may or may not be true in reality, and will have to be checked by a reader with human-level intelligence, as discussed in Section 1. The notion of semantic score, defined next, makes this idea precise.

Definition 7

Given a source \mathcal{S} and a query \mathbf{q} , the *semantic score of \mathcal{S}* (w.r.t. \mathbf{q}) is the smallest value of $\Delta(\gamma(\varepsilon(I, \mathbb{N}), F)) + \Delta(s)$ for all possible choices of F and s satisfying conditions (c1) and (c2) from Definition 6. If \mathcal{S} is not a match for \mathbf{q} , its semantic score is ∞ .

Note that a semantic score of ∞ indicates that \mathcal{S} is irrelevant to the query. Definitions 6 and 7 provide a complete definition of the reasoning task of Action-Centered IR. Given a query and a set of sources, the sources relevant to \mathbf{q} can be identified by means of Definition 6 and then ranked by relevance according to the semantic score given by Definition 7. We illustrate the process by means of the following examples.

Example 9

Continuing our running example, let us apply Definition 6 to check if \mathcal{S}_1 is a match for \mathbf{q} . Let us first look for F and s satisfying (c1). We begin with $F = \emptyset$, $s = \langle d \rangle^?$. We need to find a model π of $[\gamma(\varepsilon(I, \mathbb{N}), F), s]$ such that $\pi \models \pm\mathbf{q}$. Using the results from the previous examples, one can check that $\gamma(\varepsilon(I, \mathbb{N}), F) = \gamma(\{-m\}, \emptyset) = \{\{-m, \neg ab\}\}$ and that $[\{\{-m, \neg ab\}\}, \langle d \rangle^?]$ has a unique model $\pi = \langle \{-m, \neg ab\}, d, \{-m, \neg ab\} \rangle$. Thus, the model entails $\pm\mathbf{q}$, which means that condition (c1) is satisfied.

Next, we check condition (c2). Clearly, $\gamma(\pi_{\sigma_0} \setminus \varepsilon(I, \mathbb{N}), \emptyset) = \{\{u(m), \neg ab\}\}$ and $[\{\{u(m), \neg ab\}\}, \langle \rangle]$ has a unique model $\langle \{u(m), \neg ab\} \rangle$. The model does not entail $\pm\mathbf{q}$, and thus conditions (c2a) and (c2) are satisfied. Intuitively, this means that any assumptions made to satisfy (c1) are not directly and solely responsible for the ability of π to entail $\pm\mathbf{q}$ in (c1). Hence, it is acceptable to conclude that \mathcal{S} matches \mathbf{q} . Additionally, according to Definition 7, the semantic score of \mathcal{S}_1 is $\Delta(\gamma(\varepsilon(I, \mathbb{N}), F)) + \Delta(s) = |F| + \Delta(s) = |\emptyset| + \Delta(\langle d \rangle^?) = 0$.

Example 10

As an additional example, consider \mathcal{S}_2 , “John read a book,” from Example 1. As above, $\mathbf{q} = m$, $\mathcal{F} = \{m, ab\}$, $\mathcal{D} = \{ab\}$, $I = \emptyset$. \mathcal{E} is expanded by an additional action r , representing reading a book. The sequence of actions is captured by $\mathbb{N} = \langle r \rangle$. AD is as before.⁷

Let us begin by finding the conservative expansion $\varepsilon(I, \mathbb{N})$ through Definition 5. Similarly to the running example, $I[\mathcal{F} \setminus \mathcal{D}]$ is $\{\{m\}, \{-m\}\}$, and its elements yield completions $\{m, \neg ab\}$ and $\{-m, \neg ab\}$ respectively. Differently from Example 9, both $[\{\{m, \neg ab\}\}, \langle r \rangle^\times]$ and $[\{\{-m, \neg ab\}\},$

⁷ For simplicity, we formalize r as a no-op action.

$\langle r \rangle^\times]$ have models. Hence, $\varepsilon(I, \aleph) = \{m\} \cap \{\neg m\} = \emptyset$. That is, the story does not allow one to infer the truth value of any additional fluent.

Next, we apply Definition 6. We begin by considering the models of $[\gamma(\varepsilon(I, \aleph), F), s]$ for $F = \emptyset$, $s = \langle r \rangle^?$. One can check that $\gamma(\varepsilon(I, \aleph), F) = \gamma(\emptyset, \emptyset) = \{\{u(m), \neg ab\}\}$. $[\{\{u(m), \neg ab\}\}, \langle r \rangle^?]$ has a unique model $\pi = \langle \{u(m), \neg ab\}, r, \{u(m), \neg ab\} \rangle$. Thus, $\pi \not\models \pm \mathbf{q}$.

Another possible choice for F and s is $F = \emptyset$, $s = \langle r \rangle^\times$. One can check that $[\gamma(\varepsilon(I, \aleph), F), s]$ has two models – for instance, $\langle \{u(m), \neg ab\}, r, \{u(m), \neg ab\} \rangle$ – but neither entails $\pm \mathbf{q}$.

A more interesting choice is $F = \{m\}$, $s = \langle r \rangle^?$. Clearly, $\gamma(\varepsilon(I, \aleph), F) = \gamma(\emptyset, \{m\})$, from which it follows that $[\gamma(\varepsilon(I, \aleph), F), s]$ has two models: $\pi_1 = \langle \{m, \neg ab\}, r, \{m, \neg ab\} \rangle$ and $\pi_2 = \langle \{\neg m, \neg ab\}, r, \{\neg m, \neg ab\} \rangle$, with $\pi_1 \models \mathbf{q}$ and $\pi_2 \models \neg \mathbf{q}$ respectively. Next, we need to check condition (c2) for each. For the former, $\gamma((\pi_1)_{\sigma_0} \setminus \emptyset, \emptyset) = \{\{m, \neg ab\}\}$, and $[\{\{m, \neg ab\}\}, \langle \rangle]$ has a unique model $\langle \{m, \neg ab\} \rangle$, which entails \mathbf{q} . Since \mathbf{q} is also entailed by π_1 , (c2) is not satisfied. For π_2 , we obtain a unique model $\langle \{\neg m, \neg ab\} \rangle$, which entails $\neg \mathbf{q}$, thus failing to satisfy (c2) as well. Therefore, none of these choices for F and s yields a match. Similar conclusions can be drawn for the other choices for F and s . Hence, \mathcal{S}_2 does not match \mathbf{q} . By Definition 7, \mathcal{S}_2 has a semantic score of ∞ .

The other examples are solved similarly. We provide highlights of their solutions.

Example 2. In this example, people from countries that allow plural marriage are exceptions to the custom about first dates, and thus $I = \{ab\}$, $\aleph = \langle d \rangle$, and $I[\mathcal{F} \setminus \mathcal{D}] = \{\{m, ab\}, \{\neg m, ab\}\}$. Differently from the previous case, both sets of $I[\mathcal{F} \setminus \mathcal{D}]$ yield a model, since ab makes the executability condition inapplicable. Hence, $\varepsilon(I, \aleph) = \{ab\}$. Selecting $F = \emptyset$, $s = \langle d \rangle^?$ yields a unique model $\langle \{u(m), ab\}, d, \{u(m), ab\} \rangle \not\models \pm \mathbf{q}$. Selecting $F = \{m\}$, $s = \langle d \rangle^?$ yields two models entailing \mathbf{q} and $\neg \mathbf{q}$ respectively, but the same conclusions are entailed by $[\gamma(\pi_{\sigma_0} \setminus \varepsilon(I, \aleph), \emptyset), \langle \rangle]$, thus failing to satisfy condition (c2). Similar reasoning applies to the other cases. Because no F and s satisfying Definition 6 exist, the semantic score of \mathcal{S} is ∞ , indicating that it is irrelevant to \mathbf{q} . Note the key role played by condition (c2) in this example: without it, the source would have been deemed relevant to the query.

Example 4. In this example, AD is expanded with w causes m ; fd causes $u(m)$ and relevant executability conditions. The signature is extended accordingly. Also, $I = \emptyset$ and $\aleph = \langle d, w, fd \rangle$.

Similarly to Example 1, one can check that $\varepsilon(I, \aleph) = \{\neg m\}$. The model obtained from $F = \emptyset$ and $s = \aleph^?$ does not entail $\pm \mathbf{q}$. On the other hand, the choice of $F = \emptyset$ and $s = \langle d/\emptyset, w/\emptyset, fd/\{m\} \rangle$ yields two models, entailing \mathbf{q} and $\neg \mathbf{q}$ respectively, depending on whether m is true or false after fd . This time, condition (c2) is satisfied, since, in both cases, $\gamma(\pi_{\sigma_0} \setminus \varepsilon(I, \aleph), \emptyset) = \{\{u(m), \neg ab\}\}$ and $[\{\{u(m), \neg ab\}\}, \langle \rangle]$ does not entail $\pm \mathbf{q}$. In conclusion, \mathcal{S} indeed matches \mathbf{q} . In this case, the source has semantic score of 1, which is, as one might expect, worse than that of \mathcal{S}_1 from Example 1, while better than that of \mathcal{S}_2 .

5 Automating the Reasoning Task

In this section, we propose an approach for automating Action-Centered IR. Our approach leverages a translation of \mathcal{AL}_{IR} to ASP. The choice of ASP is motivated by the availability of well-understood mappings from action language \mathcal{AL} and its semantics to ASP, as well as of ASP-based implementations of the other modeling and reasoning techniques, discussed earlier, which we build upon. Additionally, ASP's non-monotonic nature allows one to model, in a natural and declarative way, crucial elements such as the effects of non-deterministic actions and the different

ways of formalizing uncertainty (by cases vs. explicit lack of knowledge). A brief introduction on ASP can be found in Appendix A.

5.1 ASP Implementation of the Reasoning Task

Given a consistent set I of fluent literals, a set F of fluents, a qualified action sequence s , and an action description AD , the encoding of \mathcal{AL}_{IR} is program $\Pi_{AD}(I, F, s)$, described next.

In the following, I ranges over steps in the evolution of the domain⁸; given fluent literal l , $\chi(l, I)$ stands for $holds(f, I)$ if $l = f$ and $\neg holds(f, I)$ if $l = \neg f$. The translation of a dynamic law of the form (1) depends on the form of λ . If λ is a fluent literal, the translation is:

$$\chi(\lambda, I + 1) \leftarrow occurs(e, I), \chi(l_1, I), \dots, \chi(l_n, I).$$

If λ is of the form $u(f)$, the translation of the law is:

$$\begin{aligned} u(f, I + 1) &\leftarrow occurs(e, I), \chi(l_1, I), \dots, \chi(l_n, I), \text{not } split(f, I). \\ \chi(f, I + 1) \vee \chi(\neg f, I + 1) &\leftarrow occurs(e, I), \chi(l_1, I), \dots, \chi(l_n, I), split(f, I). \end{aligned}$$

Expression $occurs(e, I)$ states that elementary action e occurs at step I in the story; $split(f, I)$ indicates that reasoning by cases should be applied to fluent f .

A state constraint of the form (2) is translated as an ASP rule of the form

$$\chi(l_0, I) \leftarrow \chi(l_1, I), \dots, \chi(l_n, I).$$

Finally, an executability condition of the form (3) is translated as a rule

$$\leftarrow occurs(e, I), \chi(l_1, I), \dots, \chi(l_n, I).$$

The translation is completed by a set of general-purpose axioms that formalize the semantics of \mathcal{AL}_{IR} . The following rules capture the notion of consistency of sets of fluents (F is a variable ranging over all fluents):

$$\leftarrow \chi(F, I), u(F, I). \quad \leftarrow \chi(\neg F, I), u(F, I).$$

Next are the inertia axioms, which are expanded in \mathcal{AL}_{IR} to accommodate extended literals:

$$\begin{aligned} \chi(F, I + 1) &\leftarrow \chi(F, I), \text{not } \chi(\neg F, I + 1), \text{not } u(F, I + 1). \\ \chi(\neg F, I + 1) &\leftarrow \chi(\neg F, I), \text{not } \chi(F, I + 1), \text{not } u(F, I + 1). \\ u(F, I + 1) &\leftarrow u(F, I), \text{not } \chi(F, I + 1), \text{not } \chi(\neg F, I + 1). \end{aligned}$$

The final set of axioms captures the definition of completion:

$$\begin{aligned} [\mathbf{g}_1] \quad &\chi(F, 0) \leftarrow \text{init}(F). \quad \chi(\neg F, 0) \leftarrow \neg \text{init}(F). \\ [\mathbf{g}_2] \quad &\chi(F, 0) \leftarrow \text{forced}(F), \text{default}(F), \text{not } \neg \text{init}(F). \\ &\chi(F, 0) \vee \chi(\neg F, 0) \leftarrow \text{forced}(F), \text{not } \text{default}(F), \\ &\quad \text{not } \text{init}(F), \text{not } \neg \text{init}(F). \\ [\mathbf{g}_3] \quad &\chi(\neg F, 0) \leftarrow \text{default}(F), \text{not } \chi(F, 0). \\ &u(F, 0) \leftarrow \text{not } \text{default}(F), \text{not } \chi(F, 0), \text{not } \chi(\neg F, 0). \end{aligned}$$

Above, atom $\text{default}(f)$ states that f is a default fluent, and is included in the form of a fact, as

⁸ We assume that the range of I is provided by the process of translating the passage to a logical representation.

a part of the translation, for every $f \in \mathcal{D}$. Atom $init(f)$ (resp., $\neg init(f)$) says that f is initially true (resp., false), i.e. is part of set I . Atom $forced(f)$ states that f is forced.

Set $[g_1]$ of rules maps the knowledge about the initial state to atoms of the form $holds(\cdot, \cdot)$. Set $[g_2]$ formalizes to the definition of forcing: the first rule ensures that a forced default fluent is set to true, and the second rule states that, when a non-default fluent is forced, both possible truth values should be considered for it. Set $[g_3]$ applies the default assumption and follows closely Definition 4: default fluents default to false, and non-default fluents default to unknown.

The next step of the definition of $\Pi_{AD}(I, F, s)$ is the encoding of its arguments. For every $f \in I$ (resp., $\neg f \in I$), $\Pi_{AD}(I, F, s)$ includes a fact $init(f)$ (resp., $\neg init(f)$). For every $f \in F$, $\Pi_{AD}(I, F, s)$ includes a fact $forced(f)$. Qualified action sequence s is encoded by a set of facts of the form $occurs(e, i)$ and $split(f, i)$, where every e and f is from s , and i is the corresponding index in the sequence of elements from s .

One can check that an expression of the form $\{e_1, \dots, e_m\}^?$ at position i of s (where each e_i is an elementary action) is translated into a collection of statements $occurs(e_1, i), \dots, occurs(e_m, i)$. An expression of the form $\{e_1, \dots, e_m\}^\times$ at position i of s is translated into a collection of statements $occurs(e_1, i), \dots, occurs(e_m, i)$ together with a statement $split(f, i)$ for every $f \in \mathcal{F}$.

This completes the definition of $\Pi_{AD}(I, F, s)$. Next, we link its answer sets to the models of $[\gamma(I, F), s]$. We say that an answer set A of a program *encodes a path* π if:

- (a) for every fluent literal l , $l \in \sigma_i$ iff $\chi(l, i) \in A$;
- (b) for every fluent f , $u(f) \in \sigma_i$ iff $u(f, i) \in A$;
- (c) for every elementary action e , $e \in a_i$ iff $occurs(e, i) \in A$;

The link between answer sets and models is established by the following.

Theorem 1

Let I be a consistent set of fluent literals, F a set of fluents, and s a qualified action sequence. A path π is a model of $[\gamma(I, F), s]$ iff there exists an answer set of $\Pi_{AD}(I, F, s)$ that encodes π .

Corollary 1

- A model π of $[\gamma(I, F), s]$ that entails a fluent literal l exists iff there exists an answer set A of $\Pi_{AD}(I, F, s)$ such that $\chi(l, k) \in A$, where k is the length of s .
- For every fluent f , $\pi \models \pm f$ iff $\{\chi(f, k), \chi(\neg f, k)\} \cap A \neq \emptyset$.

These results motivate the `FindMatch` algorithm, shown in Figure 1. Let $\|A\|$ be the number of atoms of A formed by relations *forced* and *split* (with $\|A\| = \infty$ if $A = \perp$). To illustrate the algorithm, let us trace its key parts with \mathcal{S}_1 from Example 1. Clearly, $\Pi_{AD}(I, \mathcal{F} \setminus \mathcal{D}, \aleph^\times) \supseteq \{\leftarrow occurs(d, 1), holds(m, 1), step(1). forced(m). occurs(d, 0)\}$. Step 1 infers the initial truth of fluents indirectly from the \mathcal{S}_1 , resulting in an answer set containing $\{\neg holds(m, 0), forced(m)\}$, i.e., John cannot be initially married. Hence, $I' = I \cup \{\neg m\}$. Step 4 checks condition **(c1)**. It results in a unique answer set $A \supseteq \{holds(m, 0), \neg holds(ab, 0), occurs(d, 0), \neg holds(m, 1), \neg holds(ab, 1)\}$, indicating that $\langle \{\neg m, \neg ab\}, d, \{\neg m, \neg ab\} \rangle$ entails $\pm m$. Step 4b checks condition **(c2)**. There is a single answer set $B \supseteq \{u(m, 0), \neg holds(ab, 0), u(m, 1), \neg holds(ab, 1)\}$, and, clearly, $\{holds(m, 0), \neg holds(m, 0)\} \cap B = \emptyset$. Hence, **(c2)** is satisfied and the algorithm returns A . The semantic score of \mathcal{S}_1 is $\|A\| = 0$.

The behavior of the algorithm is characterized by the following theorem, whose proof can be found in B.2.

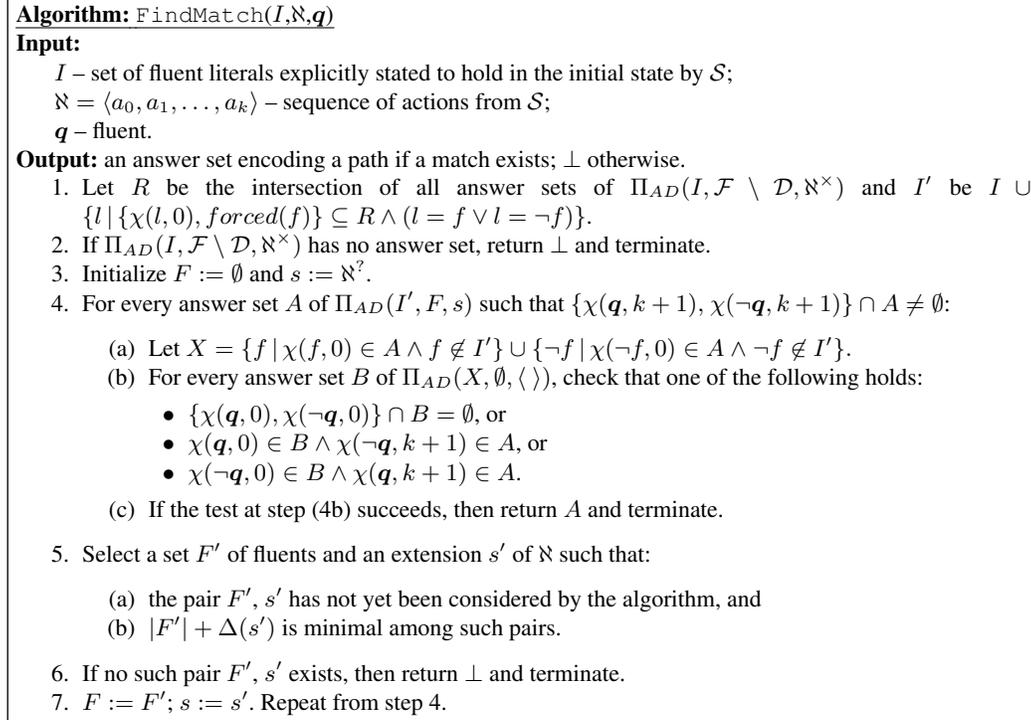


Fig. 1. FindMatch algorithm

Theorem 2

A source \mathcal{S} is a match for a query \mathbf{q} iff $\text{FindMatch}(I, \aleph, \mathbf{q}) \neq \perp$. The semantic score of \mathcal{S} is $\|\text{FindMatch}(I, \aleph, \mathbf{q})\|$.

Given a query \mathbf{q} and a collection $\mathcal{S}_1, \dots, \mathcal{S}_n$ of sources, the Action-Centered IR task of ranking the sources based on how relevant each of them is to the problem of finding an answer to \mathbf{q} can now be reduced to (a) using algorithm FindMatch to calculate $A_i = \text{FindMatch}(I_i, \aleph_i, \mathbf{q})$, where I_i and \aleph_i are the corresponding components of \mathcal{S}_i ; (b) calculating each semantic score $\|A_i\|$; and (c) sorting the sources according to their semantic scores.

5.2 Empirical Evaluation

While an exhaustive experimental evaluation is beyond the scope of this paper, we include results from a preliminary evaluation we conducted in order to assess the overall viability of our approach. The evaluation is based on a prototypical implementation of FindMatch, which can be downloaded from <http://www.mbal.tk/ACIR/>.

It follows immediately from Section 5.1 that the execution time of FindMatch only depends on the source under consideration, which means that the search over a set of sources can be trivially parallelized. Note also that the sorting of the sources based on their score is clearly dominated by the execution time of FindMatch. Thus, the time required for answering a query over a set of n sources with m identical computing resources is $t \frac{n}{m}$, where t is the average time for processing one source. As a result, in the rest of this discussion we focus on the execution of FindMatch on individual sources. We organize our evaluation along three dimensions.

Sensitivity to problem features. We evaluated the sensitivity of `FindMatch` to variations in the problem’s features by measuring its performance over 100 problem instances from the `ins-3-0` set of the Shuttle’s Reaction Control System benchmark (Balduccini et al., 2006). These instances are significant for at least a preliminary evaluation because they involve actions whose effects have intricate ramifications, and involve the practically-relevant cyberanalytics task of answering questions about a real-world cyber-physical systems.

For this part of the evaluation, we focused on stories consisting of 5 steps (an intermediate number of steps in the context of the original study), and potentially up to 3 concurrent actions per step, for a total of 15 actions. For each selected instance from the original benchmark, we randomly generated a sequence of actions of the desired length. The queries were selected in such a way that they would lead to a successful match in approximately 50% of the instances.

The results of the experiment are illustrated in Figure C 2 (see Appendix C). The figure reports the execution time for each instance, with the instances colored differently depending on whether they led to a match or not. The corresponding average times are shown as dashed lines. The execution times for instances that led to a match are substantially faster than those of instances that did not, with an average of 0.85 seconds vs. 12.81 seconds. This is not surprising, given that in the latter case the algorithm needs to explore exhaustively all possible forcings and extensions of the action sequence from the source.

Other than this distinction, it appears that performance of `FindMatch` is largely independent from the features of the source. In fact, the standard deviation for the “match” instances is 1.10 and for the “no-match” instances it is 3.71.

Sensitivity to the number of actions. Another aspect of the algorithm that we wanted to evaluate was its sensitivity to the number of actions in a source and, more specifically, to the number of time steps. Approaches to reasoning about actions and change based on ASP sometimes suffer from a rather substantial growth of the execution time as the number of steps increases. Given that the previous experiment demonstrated the overall insensitivity of the algorithm to the features of the source, for this part of the evaluation we focused on the two instances that yielded, respectively, the fastest and lowest execution times in the previous experiment.

We randomly generated sequences of actions involving a progressively increasing number of steps, ranging between 3 and 10. With up to 3 parallel actions per step, this yields sources with a number of actions between 9 and 30. The outcome of the experiment is illustrated in Figure C 3 (see Appendix C). As one might expect, the figure shows an increase of execution time as the number of steps grows. However, the increase is rather moderate, with a worst-case performance of 53.72 seconds and an average time, across all instances, of 13.08 seconds. As before, instances that yielded a match were substantially faster than those that did not. Out of 16 instances, 7 were solved in less than 1 second and 10 in less than 2 seconds (in fact, in less than 1.10 seconds).

Sensitivity to non-determinism. The final aspect of the performance of `FindMatch` that we evaluated is its scalability in the presence of actions with non-deterministic effects. As we discussed earlier, determining if a source is a match for a query may require reasoning by cases over the effects of non-deterministic actions, which tends to increase the number of iterations of `FindMatch`. For this part of the evaluation, we used the same two instances used in the previous experiment, and created 10 variants for each, so that in each variant, 2 randomly selected actions from the story were redefined to have a non-deterministic direct effects (as in the first experiment, the number of steps was 5). The choice to select 2 actions was a pragmatic one, since a source involving a large amount of uncertainty is of limited utility in the context of Action-Centered IR, since the step of manual evaluation by a human would require a substantial effort for determining

the actual effects of the actions. The results of the experiment are shown in Figure C 4 (see Appendix C). As one might expect, the execution time in this experiment was higher than in the previous ones, due to the larger number of options `FindMatch` needs to consider. The maximum execution time was 257.48 seconds and the minimum 0.81, with an average of 71.15 seconds. Out of 20 instances, 7 were solved in less than 1 second, and 9 in less than 10 seconds.

Overall considerations. A comprehensive evaluation is needed before general claims can be made, but we believe our experiments show that the approach is promising. In a complex domain such as the Reaction Control System of the Space Shuttle, our simple implementation was able to solve all instances considered in less than 260 seconds and frequently in less than 1 second.

6 Related Work

Most traditional IR systems base the relevance of a document on a syntactic measurement of the overlap of terms between query and document (Manning et al., 2008). Results using this approach may be improved via the application of query expansion (Carpineto and Ramano, 2012), an approach reformulating the original query to expand the sphere of search, for example by collecting synonyms for terms in the query and searching for documents related to those synonyms.

Several approaches for improving search results have been proposed. Recent work (Blanco and Lioma, 2012) aims at rethinking the modeling of documents by representing text as a graph whose nodes are terms linked to one another by such properties as co-occurrence in text or grammatical morphology. In this approach, the weights of connections between terms are learned using graph search algorithms such as PageRank (Page et al., 1999). Another interesting area of related research is Temporal IR, or T-IR. Work in this field aims to improve the results of Information Retrieval methods by extracting and leveraging temporal information in both documents and queries. Campos (2015) presents an extensive survey of the topic. Approaches involving semantic networks, such as Google’s Knowledge Graph, bolster IR techniques with world facts and relationships. However, they are not concerned with a deeper analysis of query and documents.

There are a number of research efforts which, while not directly comparable to the work presented here, demonstrate the numerous ways in which IR and complex reasoning tasks are being addressed. One remarkable line of research is that of the question answering agent architecture by Mitra and Baral (2016). In response to the Facebook set of pre-requisite toy tasks for intelligent question answering (Weston et al., 2015), their architecture features a sophisticated reasoning layer that leverages Inductive Logic Programming, implemented in ASP, to learn the knowledge needed to answer the toy task questions. The authors demonstrated that their agent either matches or outperforms machine learning approaches on the Facebook dataset. It is important to note that this technique is aimed at question answering, not IR. However, the research on the question answering agent architecture demonstrates the advantages of leveraging formal reasoning for these kinds of tasks, and provides an encouraging indication for our work as well.

Another approach based on logic and reasoning is in (Lukasiewicz and Straccia, 2007), where the authors aim to answer vague queries such as “find a car that costs around \$11,000 with about 15,000 miles” by leveraging description logics and logic programming to rank potential answers by a defined degree of relatedness. Although the notion of degree of relatedness bears some superficial similarities with our work, it should be noted that, once again, this approach is focused on question answering rather than IR. Another major difference is that our work aims at reasoning about sequences of events and the effects of those events, both direct and indirect.

Liu et al. (2007) presents a novel benchmark dataset for the evaluation of Machine Learning

algorithms for ranking text sources in IR. Citing a growing field of feature-based ranking for IR, the authors identify and address the lack of standard benchmarks. Although not directly related to our approach, this work may offer useful leads for the creation of evaluation benchmarks.

Finally, Dong et al. (2014) propose an approach for the creation of knowledge bases about actions and their effects. They leverage the process of *knowledge fusion*, in which large-scale knowledge bases are automatically extracted from text and associated with a quality measure.

7 Conclusions and Future Work

In this paper, we presented an investigation of an IR task in which sources containing sequences of events are matched to a query about the state of the world after those events. While this task is critical to simplifying access to information and reducing information overload, traditional IR techniques appear unfit to solve it. Thus, we began by analyzing the problem from a common-sensical and intuitive perspective, and identified characteristics of the corresponding reasoning tasks. We focused particularly on the ability to carry out the fine-grained reasoning needed for a determination of relevance in the presence of uncertainty. Our investigation led us to developing a novel action language, which we used to give an accurate definition of the Action-Centered IR task. Finally, we defined an ASP-based procedure for automating the reasoning task and conducted an empirical evaluation of its scalability.

At this stage of our research, we have focused on the definition and study of the core IR task. Future work will address the connection with natural language processing and with available knowledge repositories, the development of an end-to-end system, and the quantitative evaluation of our approach. Additionally, it will be interesting to study particular classes of query-source pairs for which simplified forms of reasoning may be possible. For instance, one can check that sources that can be formalized by a deterministic action description without default fluents may be processed without the need for reasoning by cases and, in fact, yield only two possible semantic scores for any query: 0 and ∞ . Identifying additional classes may lead to a better understanding of the problem and to more efficient computations.

References

- BALDUCCINI, M. AND GELFOND, M. 2003. Diagnostic reasoning with A-Prolog. *Journal of Theory and Practice of Logic Programming (TPLP)* 3, 4–5 (Jul), 425–461.
- BALDUCCINI, M., GELFOND, M., AND NOGUEIRA, M. 2006. Answer Set Based Design of Knowledge Systems. *Annals of Mathematics and Artificial Intelligence* 47, 1–2, 183–219.
- BARAL, C. AND GELFOND, M. 2000. Reasoning Agents In Dynamic Domains. In *Workshop on Logic-Based Artificial Intelligence*. Kluwer Academic Publishers, 257–279.
- BLANCO, R. AND LIOMA, C. 2012. Graph-Based Term Weighting for Information Retrieval. *Information Retrieval* 15.1, 54–92.
- CAMPOS, R. 2015. Survey of Temporal Information Retrieval and Related Applications. *ACM Computing Surveys (CSUR)* 47, 2.
- CARPINETO, C. AND RAMANO, G. 2012. A Survey of Automatic Query Expansion in Information Retrieval. *ACM Computing Surveys (CSUR)* 44, 1.
- DONG, X. L., GABRILOVICH, E., HEITZ, G., HORN, W., MURPHY, K., SUN, S., AND ZHANG, W. 2014. From data fusion to knowledge fusion. *Proceedings of the VLDB Endowment* 7, 10, 881–892.

- GELFOND, M. AND LIFSCHITZ, V. 1991. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing* 9, 365–385.
- GLAVAS, G. AND SNAJDER, J. 2014. Event Graphs for Information Retrieval and Multi-Document Summarization. *Expert Systems with Applications* 41, 15, 6904–6916.
- INCLEZAN, D. 2016. CoreALMlib: An ALM Library Translated from the Component Library. In *32nd International Conference on Logic Programming (ICLP16)*.
- KORFHAGE, R. R. 1997. *Information Storage and Retrieval*. John Wiley and Sons, Inc.
- LEBLANC, E. AND BALDUCCINI, M. 2016. Interpreting Natural Language Sources Using Transition Diagrams. In *Logic Programming with Constraints for Language Processing (CSLP2016)*, H. Christiansen and V. Dahl, Eds.
- LIERLER, Y., INCLEZAN, D., AND GELFOND, M. 2017. Action Languages and Question Answering. In *12th International Conference on Computational Semantics (IWCS 2017)*.
- LIFSCHITZ, V. AND TURNER, H. 1994. Splitting a logic program. In *Proceedings of the 11th International Conference on Logic Programming (ICLP94)*. 23–38.
- LIU, T.-Y., XU, J., QIN, T., XIONG, W., AND LI, H. 2007. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *Proceedings of SIGIR 2007 workshop on learning to rank for information retrieval*. Vol. 310. ACM Amsterdam, The Netherlands.
- LUKASIEWICZ, T. AND STRACCIA, U. 2007. Top-k retrieval in description logic programs under vagueness for the semantic web. In *International Conference on Scalable Uncertainty Management*. Springer, 16–30.
- MANNING, C., CHRISTOPHER, D., RAGHAVAN, P., AND SCHÜTZE, H. 2008. *Introduction to Information Retrieval*. Vol. 1. Cambridge University Press.
- MAREK, V. W. AND TRUSZCZYNSKI, M. 1999. *The Logic Programming Paradigm: a 25-Year Perspective*. Springer Verlag, Berlin, Chapter Stable Models and an Alternative Logic Programming Paradigm, 375–398.
- MATUSZEK, C., CABRAL, J., WITBROCK, M. J., AND DEOLIVEIRA, J. 2006. An Introduction to the Syntax and Content of CYC. In *AAAI Spring Symposium: Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*. 44–49.
- MITRA, A. AND BARAL, C. 2016. Addressing a question answering challenge by combining statistical methods with inductive rule learning and reasoning. In *AAAI*. 2779–2785.
- MORALES, R., TU, P. H., AND SON, T. C. 2007. An Extension to Conformant Planning Using Logic Programming. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI’07)*, M. M. Veloso, Ed. 1991–1996.
- NGUYEN, V., MITRA, A., AND BARAL, C. 2015. The NL2KR Platform for Building Natural Language Translation Systems. In *53rd Annual Meeting of the Association for Computational Linguistics (ACL-IJCNLP 2015)*. 899–908.
- PAGE, L., BRIN, S., MOTWANI, R., AND WINOGRAD, T. 1999. The pagerank citation ranking: bringing order to the web.
- SUCHANEK, F. M., KASNECI, G., AND WEIKUM, G. 2008. Yago: A Large Ontology from Wikipedia and WordNet. *Web Semantics: Science, Services and Agents on the World Wide Web* 6, 3, 203–217.
- WESTON, J., BORDES, A., CHOPRA, S., RUSH, A. M., VAN MERRIËNBOER, B., JOULIN, A., AND MIKOLOV, T. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*.

Appendix A Answer Set Programming

ASP (Gelfond and Lifschitz, 1991; Marek and Truszczyński, 1999) is a knowledge representation language with roots in the research on the semantics of logic programming languages and non-monotonic reasoning. The syntax of the language is defined as follows.

Let Σ be a signature containing constant, function and predicate symbols. Terms and atoms are formed as in first-order logic. A *literal* is an atom a or its negation $\neg a$. A *rule* is a statement of the form:

$$h_1 \vee h_2 \dots \vee h_k \leftarrow l_1, \dots, l_m, \text{not } l_{m+1}, \dots, \text{not } l_n \quad (\text{A1})$$

where each h_i and l_i is a literal and *not* is called *default negation* operator. The intuitive meaning of A1 is given in terms of a rational agent reasoning about its own beliefs and it is summarized by the statement “a rational agent that believes l_1, \dots, l_m and has no reason to believe l_{m+1}, \dots, l_n , must believe one of h_1, \dots, h_k .” If $m = n = 0$, symbol \leftarrow is omitted and the rule is a *fact*. Rules of the form $\perp \leftarrow l_1, \dots, \text{not } l_n$ are abbreviated $\leftarrow l_1, \dots, \text{not } l_n$, and called *constraints*, intuitively meaning that $\{l_1, \dots, \text{not } l_n\}$ must not be satisfied. A rule with variables is interpreted as a shorthand for the set of rules obtained by replacing the variables with all possible variable-free terms. A *program* is a set of rules over Σ .

Next, we define the semantics of ASP. We say that a consistent set S of literals is closed under a rule if $\{h_1, \dots, h_k\} \cap S \neq \emptyset$ whenever $\{l_1, \dots, l_m\} \subseteq S$ and $\{l_{m+1}, \dots, l_n\} \cap S = \emptyset$. Set S is an answer set of a *not-free* program Π if S is the minimal set closed under its rules. The reduct, Π^S , of a program Π w.r.t. S is obtained from Π by removing every rule containing an expression “not l ” s.t. $l \in S$ and by removing every other occurrence of not l . Set S is an answer set of Π if it is the answer set of Π^S .

Appendix B Proofs of Theorems

In this appendix, we provide proofs of the main results of this paper.

B.1 Proof of Theorem 1

Before we proceed to the proof of Theorem 1, we need to introduce the following notions. Let AD be an action description of \mathcal{AL}_{IR} , n be a positive integer, and $\Sigma(AD)$ be the signature of AD . $\Sigma^n(AD)$ denotes the signature obtained as follows:

- $const(\Sigma^n(AD)) = const(\Sigma(AD)) \cup \{0, \dots, n\}$
- $pred(\Sigma^n(AD)) = \{holds, u, split, occurs\}$

Let

$$\alpha^n(AD) = \langle \Sigma^n(AD), \Pi^\alpha(AD) \rangle, \quad (\text{B1})$$

where

$$\Pi^\alpha(AD) = \bigcup_{r \in AD} \alpha(r), \quad (\text{B2})$$

and $\alpha(r)$ is defined as follows:

- $\alpha(e \text{ causes } \lambda \text{ if } l_1, \dots, l_n) \text{ is}$

$$\chi(\lambda, I + 1) \leftarrow occurs(e, I), \chi(l_1, I), \dots, \chi(l_n, I). \quad (\text{B3})$$

if λ is a fluent literal. If λ is of the form $u(f)$, the translation of the law is

$$u(f, I + 1) \leftarrow \text{occurs}(e, I), \chi(l_1, I), \dots, \chi(l_n, I), \text{not } \text{split}(f, I). \quad (\text{B4})$$

$$\chi(f, I + 1) \vee \chi(\neg f, I + 1) \leftarrow \text{occurs}(e, I), \chi(l_1, I), \dots, \chi(l_n, I), \text{split}(f, I). \quad (\text{B5})$$

- $\alpha(l_0 \text{ if } l_1, \dots, l_n)$ is

$$\chi(l_0, T) \leftarrow \chi(l_1, T), \dots, \chi(l_n, T). \quad (\text{B6})$$

- $\alpha(e \text{ impossible if } l_1, \dots, l_n)$ is

$$\leftarrow \chi(l_1, T), \dots, \chi(l_n, T), \text{occurs}(e, T).$$

Let also

$$\Phi^n(AD) = \langle \Sigma^n(AD), \Pi^\Phi(AD) \rangle, \quad (\text{B7})$$

where

$$\Pi^\Phi(AD) = \Pi^\alpha(AD) \cup \Pi \quad (\text{B8})$$

and Π contains the following rules:

$$\chi(F, I + 1) \leftarrow \chi(F, I), \text{not } \chi(\neg F, I + 1), \text{not } u(F, I + 1). \quad (\text{B9})$$

$$\chi(\neg F, I + 1) \leftarrow \chi(\neg F, I), \text{not } \chi(F, I + 1), \text{not } u(F, I + 1). \quad (\text{B10})$$

$$u(F, I + 1) \leftarrow u(F, I), \text{not } \chi(F, I + 1), \text{not } \chi(\neg F, I + 1). \quad (\text{B11})$$

Π also contains the following rules:

$$\leftarrow \chi(F, I), u(F, I). \quad (\text{B12})$$

$$\leftarrow \chi(\neg F, I), u(F, I). \quad (\text{B13})$$

When we refer to a single action description, we drop argument AD from the above expressions.

For the rest of this section, we will focus on ground programs. In order to keep notation simple, we will use α^n and Φ^n to denote the ground versions of the programs previously defined.

The following notation will be useful in our further discussion. Given a time point t , a state σ , and a compound action a , let

$$\begin{aligned} \chi(\sigma, t) &= \{ \chi(l, t) \mid l \in \sigma \cap Lit \} \cup \\ &\quad \{ u(f, t) \mid u(f) \in \sigma \} \\ \text{occurs}(a, t) &= \{ \text{occurs}(e, t) \mid e \in a \} \end{aligned} \quad (\text{B14})$$

These sets can be viewed as the representation of σ and a in ASP. Let also

$$\text{split}(q_t, t) = \{ \text{split}(f, t) \mid f \in q_t \}$$

which represents a set of fluents to which reasoning by cases should be applied according to a qualifier q_t .

For any action description AD , state σ_0 , and qualified action sequence $s = \langle a_0/q_0, \dots, a_{n-1}/q_{n-1} \rangle$, let $\Phi^n(\sigma_0, s)$ denote

$$\Phi^n \cup \{ \text{occurs}(a_i, i) \mid a_i \text{ is in } s \} \cup \{ \text{split}(q_i, i) \mid q_i \text{ is in } s \} \quad (\text{B15})$$

Where possible, we drop the first argument, and denote the program by $\Phi^n(\sigma_0, s)$. Also, for convenience, we write $\Phi^1(\sigma_0, a_0, q_0)$ when $n = 1$.

An important property of Cn_Z that we will use later is:

Lemma 3

For every fluent f , $u(f) \in Cn_Z$ iff $u(f) \in S$.

Proof

The thesis follows trivially from the observation that proper extended literals do not occur in state constraints. \square

The following lemma will be helpful in proving the main result of this section. It states the correspondence between (single) transitions of the transition diagram and answer sets of the corresponding ASP program.

Lemma 4

Let AD be an action description and $\mathcal{T}(AD)$ be the transition diagram it describes. Then, $\langle \sigma_0, a_0, \sigma_1 \rangle \in \mathcal{T}(AD)$ iff $\sigma_1 = \{l \mid \chi(l, 1) \in A\} \cup \{u(f) \mid u(f, 1) \in A\}$ for some qualifier q_0 and some answer set A of $\Phi^1(\sigma_0, a_0, q_0)$.

Proof

Let us define

$$Y_{\sigma_0, a_0, q_0} = \chi(\sigma_0, 0) \cup \text{occurs}(a_0, 0) \cup \text{split}(q_0, 0) \quad (\text{B16})$$

and

$$\Phi^1(\sigma_0, a_0, q_0) = \Phi^1 \cup Y_{\sigma_0, a_0, q_0}$$

Left-to-right. Let us construct the qualifier q_0 as:

$$\begin{aligned} q_0 = \{f \mid & e \text{ causes } u(f) \text{ if } \Gamma \in AD, \\ & e \in a_0, \Gamma \subseteq \sigma_0, \text{ and} \\ & u(f) \notin \sigma_1\} \end{aligned} \quad (\text{B17})$$

The set q_0 is an ASP representation of a qualifier q_0 in a qualified action sequence.

Let us show that, if $\langle \sigma_0, a_0, \sigma_1 \rangle \in \mathcal{T}(AD)$, then

$$A = Y_{\sigma_0, a_0, q_0} \cup \chi(\sigma_1, 1) \quad (\text{B18})$$

is an answer set of $\Phi^1(\sigma_0, a_0, q_0)$. Notice that $\langle \sigma_0, a_0, \sigma_1 \rangle \in \mathcal{T}(AD)$ implies that σ_1 is a state. Herein, we refer to $\Phi^1(\sigma_0, a_0, q_0)$ as P .

Let us prove that A is the minimal set of literals closed under the rules of the reduct P^A . Let $\mathbb{N}^{\alpha^1(AD)}$ be the set of rules of $\alpha^1(AD)$ of form (B4). P^A contains:

- a) set Y_{σ_0, a_0, q_0} .
- b) all rules in $\alpha^1(AD) \setminus \mathbb{N}^{\alpha^1(AD)}$.
- c) a rule

$$u(f, 1) \leftarrow \text{occurs}(e, 0), \chi(l_1, 0), \dots, \chi(l_n, 0).$$

for every fluent f such that $\text{split}(f, 0) \notin A$.

- d) a rule

$$\chi(l, 1) \leftarrow \chi(l, 0)$$

for every fluent literal l such that $\chi(l, 1) \in A$ and a rule

$$\chi(-l, 1) \leftarrow \chi(-l, 0)$$

for every fluent literal $-l$ such that $\chi(-l, 1) \in A$.

e) a rule

$$u(f, 1) \leftarrow u(f, 0)$$

for every fluent f such that $u(f, 1) \in A$.

Note that because A is an answer set, $\chi(f, 1) \in A \Leftrightarrow \chi(\neg f, 1) \notin A$ and $u(f, 1) \notin A$. The conditions for $\chi(\neg f) \in A$ and $u(f) \in A$ can be similarly described.

A is closed under P^A . We will prove it for every rule of the program.

1. Rules of groups (a), (d), and (e): obvious.
2. Rules of group (b) encoding dynamic laws of the form e causes λ if l_1, \dots, l_n when λ is a fluent literal:

$$\chi(\lambda, 1) \leftarrow \text{occurs}(e, 0), \chi(l_1, 0), \dots, \chi(l_n, 0).$$

If $\{\text{occurs}(e, 0), \chi(l_1, 0), \dots, \chi(l_n, 0)\} \subseteq A$, then, by (B18), $\{l_1, \dots, l_n\} \subseteq \sigma_0$ and $e \in a_0$. Therefore, the preconditions of the dynamic law are satisfied by σ_0 . Hence (4) implies $\lambda \in \sigma_1$. By (B18), $\chi(\lambda, 1) \in A$.

3. Rules of group (b) encoding dynamic laws of the form e causes λ if l_1, \dots, l_n when λ is of the form $u(f)$:

$$\chi(f, 1) \vee \chi(\neg f, 1) \leftarrow \text{occurs}(e, 0), \chi(l_1, 0), \dots, \chi(l_n, 0), \text{split}(f, 0).$$

Let us suppose that $\text{split}(f, 0) \in A$. In fact, if that is not the case, then A is trivially closed under the rule. Similarly, assume $\{\text{occurs}(e, 0), \chi(l_1, 0), \dots, \chi(l_n, 0)\} \subseteq A$. Then, by construction of Y_{σ_0, a_0, q_0} , $\text{split}(f, 0) \in \text{split}(q_0, 0)$. In turn, by construction of $\text{split}(q_0, 0)$ and from (B17) we conclude that $f \in q_0$ and that $u(f) \notin \sigma_1$. Because σ_1 is complete from (5), we conclude that either f or $\neg f$ is in σ_1 . By (B18), either $\chi(f, 1) \in A$ or $\chi(\neg f, 1) \in A$.

4. Rules of group (b) encoding state constraints of the form l_0 if l_1, \dots, l_n :

$$\chi(l_0, t) \leftarrow \chi(l_1, t), \dots, \chi(l_n, t).$$

If $\{\chi(l_1, t), \dots, \chi(l_n, t)\} \subseteq A$, then, by (B18), $\{l_1, \dots, l_n\} \subseteq \sigma_t$, i.e. the preconditions of the state constraint are satisfied by σ_t . If $t = 1$, then (5) implies $l_0 \in \sigma_1$. By (B18), $\chi(l_0, t) \in A$. If $t = 0$, since states are closed under the state constraints of AD , we have that $l \in \sigma_0$. Again by (B18), $\chi(l_0, t) \in A$.

5. Rules of group (b) encoding executability conditions of the form e impossible_if l_1, \dots, l_n :

$$\leftarrow \text{occurs}(e, 0), \chi(l_1, 0), \dots, \chi(l_n, 0).$$

Since $\langle \sigma_0, a_0, \sigma_1 \rangle \in \mathcal{T}(AD)$ by hypothesis, $\langle \sigma_0, a_0 \rangle$ does not satisfy the preconditions of any executability condition. Then, either $e \notin a_0$ or $l_i \notin \sigma_0$ for some i . By (B18), the body of this rule is not satisfied.

6. Rules of group (c) encoding dynamic laws when λ is of the form $u(f)$:

$$u(f, 1) \leftarrow \text{occurs}(e, 0), \chi(l_1, 0), \dots, \chi(l_n, 0).$$

If the rule is in P^A , then $\text{split}(f, 0) \notin A$. By construction of Y_{σ_0, a_0, q_0} , $\text{split}(f, 0) \notin \text{split}(q_0, 0)$. By construction of $\text{split}(q_0, 0)$, $f \notin q_0$ and from (B17) it follows that $u(f) \in \sigma_1$. By (B18), $u(f, 1) \in A$.

A is the minimal set closed under the rules of P^A . We will prove this by assuming that there exists a set $B \subseteq A$ such that B is closed under the rules of P^A , and by showing that $B = A$.

First of all,

$$Y_{\sigma_0, a_0, q_0} \subseteq B, \quad (\text{B19})$$

since these are facts in P^A .

Let

$$\delta = \{l \mid \chi(l, 1) \in B\}. \quad (\text{B20})$$

Since $B \subseteq A$,

$$\delta \subseteq \sigma_1 \quad (\text{B21})$$

Let W be the element of $\mathbb{E}(a_0, \sigma_0)$ satisfying (4). We will show that $\delta = \sigma_1$ by proving that

$$\delta = CN_Z(W \cup (\sigma_1 \cap \sigma_0)). \quad (\text{B22})$$

Dynamic laws. Let d be a dynamic law of AD of the form e causes λ if l_1, \dots, l_n , such that $e \in a_0$ and $\{l_1, \dots, l_n\} \subseteq \sigma_0$. Because of (B19), $\chi(\{l_1, \dots, l_n\}, 0) \subseteq B$ and $o(e, 0) \in B$. If λ is a fluent literal, then since B is closed under $\alpha(d)$, $\chi(\lambda, 1) \in B$, and $\lambda \in \delta$. Therefore, $W \subseteq \delta$. It can be similarly shown if λ is a properly extended literal.

Inertia. P^A contains a (reduced) inertia rule of the form

$$\chi(f, 1) \leftarrow \chi(f, 0). \quad (\text{B23})$$

for every fluent $f \in \sigma_1$. Suppose $l \in \sigma_1 \cap \sigma_0$. Then, $\chi(l, 0) \in Y_{\sigma_0, a_0, q_0}$, and, since B is closed under (B23), $\chi(f, 1) \in B$. Therefore, $\sigma_1 \cap \sigma_0 \subseteq \delta$. The same argument applies to the other reduced inertia rules.

State constraints. Let r be a state constraints of AD of the form l_0 if l_1, \dots, l_n , such that

$$\chi(\{l_1, \dots, l_n\}, 0) \subseteq B. \quad (\text{B24})$$

Since B is closed under $\alpha(r)$, $\chi(l_0, 1) \in B$, and $l_0 \in \delta$. Then, δ is closed under the state constraints of AD .

Summing up, (B22) holds. From (4) and (B21), we obtain $\sigma_1 = \delta$. Therefore $\chi(\sigma_1, 1) \subseteq B$.

At this point we have shown that $Y_{\sigma_0, a_0, q_0} \cup \chi(\sigma_1, 1) \subseteq B \subseteq A$.

Right-to-left. Let A be an answer set of P and let $\sigma_1 = \{l \mid \chi(l, 1) \in A\} \cup \{u(f) \mid u(f, 1) \in A\}$. We have to show that

$$\sigma_1 = CN_Z(W \cup (\sigma_1 \cap \sigma_0)) \text{ for some } W \in \mathbb{E}(a_0, \sigma_0) \quad (\text{B25})$$

as well as that $\langle \sigma_0, a_0 \rangle$ respects all executability conditions and that σ_1 is consistent and complete.

σ_1 consistent. Obvious, since A is a (consistent) answer set by hypothesis.

σ_1 complete. By contradiction, and without loss of generality, let f be a fluent s.t. $f \notin \sigma_1$, $\neg f \notin \sigma_1$, $u(f) \notin \sigma_1$, and $f \in \sigma_0$ (since σ_0 is complete by hypothesis, if $f \notin \sigma_0$, we can still select $\neg f$ or $u(f)$). Then, the reduct P^A contains a rule

$$\chi(f, 1) \leftarrow \chi(f, 0). \quad (\text{B26})$$

Since A is closed under P^A , $\chi(f, 1) \in A$ and $f \in \sigma_1$. Contradiction.

Executability conditions respected. By contradiction, assume that law r of form e impossible if

l_1, \dots, l_n is not respected. Note that $\chi(\{l_1, \dots, l_n\}, 0) \subseteq A$ and $occurs(e, 0) \in A$. Therefore, the body of $\alpha(r)$ is satisfied by A , and A is not an answer set.

(B25) holds. Let us construct W so that:

- $W \supseteq E(a_0, \sigma_0) \cap Lit$
- for every $u(f) \in E(a_0, \sigma_0)$:
 - if $f \notin q_0$, then $u(f) \in W$
 - otherwise, $f \in W$ if $\chi(f, 1) \in A$ and $\neg f \in W$ if $\chi(\neg f, 1) \in A$.

One can check that $W \in \mathbb{E}(a_0, \sigma_0)$.

Next, let us prove that $\sigma_1 \supseteq W$, i.e. that for every $\lambda \in W, \lambda \in \sigma_1$. Suppose $\lambda \in E(a_0, \sigma_0) \cap Lit$. There must exist a dynamic law d of the form (1) such that $\{l_1, \dots, l_n\} \subseteq \sigma_0$ and $e \in a_0$. Since A is closed under (B3) of $\alpha(d)$, it follows that $\chi(\lambda, 1) \in A$. By construction of $\sigma_1, \lambda \in \sigma_1$.

Let us now consider the case in which $\lambda \notin E(a_0, \sigma_0) \cap Lit$. There must be a dynamic law d of the form e causes $u(f)$ if l_1, \dots, l_n such that f is the fluent that occurs in λ . It must be the case that $\{l_1, \dots, l_n\} \subseteq \sigma_0$, and $e \in a_0$. Note that either $f \in q_0$ or $f \notin q_0$.

If $f \notin q_0$, then by construction of W it must be the case that λ is $u(f)$. Let us consider (B4) from $\alpha(d)$. Because A is closed under it, it follows that $u(f, 1) \in A$. By construction of σ_1 , we conclude that $u(f) \in \sigma_1$.

Next, consider the case in which $f \in q_0$. If λ is f , then by construction of W , one can conclude that $\chi(f, 1) \in A$. It follows, then, that $f \in \sigma_1$. If λ is $\neg f$, with similar reasoning we derive that $\neg f \in \sigma_1$. This concludes the proof that $\sigma_1 \supseteq W$.

Additionally, $\sigma_1 \supseteq \sigma_1 \cap \sigma_0$ is trivially true.

Let us prove that σ_1 is closed under the state constraints of AD . Consider a state constraint s , of the form l_0 if l_1, \dots, l_n , such that $\{l_1, \dots, l_n\} \subseteq \sigma_0$. Since A is closed under $\alpha(s)$, $\chi(l_0, 1) \in A$. By construction of $\sigma_1, l_0 \in \sigma_1$.

Let us prove that σ_1 is the minimal set satisfying all conditions. By contradiction, assume that there exists a set $\delta \subset \sigma_1$ such that $\delta \supseteq W \cup (\sigma_1 \cap \sigma_0)$ and that δ is closed under the state constraints of AD . We will prove that this implies that A is not an answer set of P .

Let A' be the set obtained by removing from A all literals $\chi(l, 1)$ such that $l \in \sigma_1 \setminus \delta$ and all atoms of form $u(f, 1)$ such that $u(f) \in \sigma_1 \setminus \delta$. Since $\delta \subset \sigma_1, A' \subset A$.

Since $\delta \supseteq W \cup (\sigma_1 \cap \sigma_0)$, for every extended fluent literal $\lambda \in \sigma_1 \setminus \delta$ it must be true that $\lambda \notin \sigma_0$ and $\lambda \notin W$. From Lemma 3, we conclude that λ must be a fluent literal. Therefore there must exist (at least) one state constraint λ if l_1, \dots, l_n such that $\{l_1, \dots, l_n\} \subseteq \sigma_1$ and $\{l_1, \dots, l_n\} \not\subseteq \delta$. Hence, A' is closed under the rules of P^A . This proves that A is not an answer set of P . Contradiction. \square

Corollary 2

Let AD be an action description and $\mathcal{T}(AD)$ be the transition diagram it describes. Then, $\langle \sigma_0, a_0, \sigma_1, \dots, a_{n-1}, \sigma_n \rangle$ is a path of $\mathcal{T}(AD)$ iff there exist qualifiers q_0, q_1, \dots, q_{n-1} and an answer set A of $\Phi^n(\sigma_0, \langle a_0/q_0, a_1/q_1, \dots, a_{n-1}/q_{n-1} \rangle)$ such that, for every $1 \leq i \leq n$, $\sigma_i = \{l \mid \chi(l, i) \in A\} \cup \{u(f) \mid u(f, i) \in A\}$.

Proof

The thesis can be easily proven by induction from Lemma 4. \square

Theorem 1

Let I be a consistent set of fluent literals, F be a set of fluents, and s be a qualified action

sequence. A path π is a model of $[\gamma(I, F), s]$ iff there exists an answer set of $\Pi_{AD}(I, F, s)$ that encodes π .

Proof

The proof leverages Corollary 2 and the Splitting Set Lemma (Lifschitz and Turner, 1994). First of all, note that it is possible to split $\Pi_{AD}(I, F, s)$ in such a way that the bottom corresponds to rules $[\mathbf{g}_1]$, $[\mathbf{g}_2]$, $[\mathbf{g}_3]$ (see Section 5.1) together with facts encoding I and F , as well as rules encoding the state constraints for time step 0. One can check that the answer sets of the bottom encode the completion $\gamma(I, F)$, and that every element of $\gamma(I, F)$ is a state of $\tau(AD)$.

The thesis follows from the application of Corollary 2 to each $\sigma_0 \in \gamma(I, F)$, after noticing the correspondence between the top of $\Pi_{AD}(I, F, s)$ and program $\Phi^n(\sigma_0, s)$. \square

B.2 Proof of Theorem 2

Theorem 2

A source \mathcal{S} is a match for a query \mathbf{q} iff $\text{FindMatch}(I, \aleph, \mathbf{q}) \neq \perp$. The semantic score of \mathcal{S} is $|\text{FindMatch}(I, \aleph, \mathbf{q})|$.

Proof

We begin by showing that the algorithm terminates. This follows simply from the consideration that, in the worst case, the algorithm proceeds to a systematic enumeration of the subsets of \mathcal{F} and of the extensions of \aleph (refer to steps (3), (5), and (7)), which are clearly finite, and terminates when all have been enumerated (step (6)).

Next, we demonstrate that if $\Pi_{AD}(I, \mathcal{F} \setminus \mathcal{D}, \aleph^\times)$ has at least one answer set, then step (1) of the algorithm finds $\varepsilon(I, \aleph)$, i.e. that $I' = \varepsilon(I, \aleph)$. Note that the existence of an answer set is verified at step (2).

Left-to-right. Let A be an answer set of $\Pi_{AD}(I, \mathcal{F} \setminus \mathcal{D}, \aleph^\times)$. From Theorem 1, it follows that A encodes a model π_A of $[\gamma(I, \mathcal{F} \setminus \mathcal{D}), \aleph^\times]$. By construction of $\gamma(I, \mathcal{F} \setminus \mathcal{D})$,

$$\text{there exists } I' \in I[\mathcal{F} \setminus \mathcal{D}] \text{ such that } \pi_A \text{ is a model of } [\gamma(I'), \aleph^\times]. \quad (\text{B27})$$

Note that $l \in I'$ iff $l \in I$ or $I \in \{l' \mid \{\chi(l', 0), \text{forced}(l'_f)\} \subseteq R, \text{ where } l'_f \text{ is the fluent from which } l' \text{ is formed. If } l \in I, \text{ then from Proposition 2, } l \in \varepsilon(I, \aleph) \text{ and the thesis is proven from the observation that the hypothesis of existence of an answer set guarantees the existence of } \varepsilon(I, \aleph). \text{ In the other case, it follows that } \chi(l, 0) \in R \text{ and that } \text{forced}(l_f) \in R. \text{ From the former and (B27), it follows that } l \in \bigcap_{Y \in I[\mathcal{F} \setminus \mathcal{D}]} \gamma(Y). \text{ Hence,}$

$$l \in \gamma(Y) \text{ for every } Y \in I[\mathcal{F} \setminus \mathcal{D}]. \quad (\text{B28})$$

By construction of $\Pi_{AD}(I, \mathcal{F} \setminus \mathcal{D}, \aleph^\times)$, $\text{forced}(l_f) \in R$ iff $l_f \in \mathcal{F} \setminus \mathcal{D}$. By definition of forcing of a fluent, every element of $I[\mathcal{F} \setminus \mathcal{D}]$ contains either l or \bar{l} . From Proposition 1, $\gamma(Y)$ is consistent and includes Y . From (B28) and the fact that $l \in \gamma(Y)$, it follows that $l \in Y$. Hence, $l \in \bigcap_{Y \in I[\mathcal{F} \setminus \mathcal{D}]} Y$ and thus $l \in \varepsilon(I, \aleph)$. From the generality of l , it follows that $I' = \varepsilon(I, \aleph)$.

Right-to-left. The conclusion follows from Definition 5 and Theorem 1 in a straightforward way.

Next, we prove that the algorithm terminates at step (4c) iff \mathcal{S} is a match for \mathbf{q} . From Theorem 1, Corollary 1, and from our observations about step (1), it follows that for every answer set A found at step (4), there exists a model π_A of $[\gamma(\varepsilon(I, \aleph), F), s]$ that encodes A and satisfies condition (c1) of Definition 6. With similar considerations, one can conclude that for every answer

set B of $\Pi_{AD}(X, \emptyset, \langle \rangle)$ there exists a model π_B of $[\gamma(\pi_{\sigma_0} \setminus \varepsilon(I, \aleph), \emptyset), \langle \rangle]$, where π_{σ_0} is defined in Definition 6. Using Corollary 1, one can check that the three tests at step (4b) ensure that condition (c2) from Definition 6 is satisfied by π_A and π_B . Thus, if the algorithm terminates at step (4c), then \mathcal{S} is a match for q .

The right-to-left direction is proven by contradiction. We assume that the algorithm never reaches step (4c), and yet \mathcal{S} is a match for q . From Definition 6, it follows that there exist π and π' satisfying conditions (c1) and (c2). From Theorem 1 and earlier considerations, it follows that there exist answer sets A and B satisfying the conditions from step (4) of the algorithm. This means that the condition of the *if* statement at step (4c) is true, and thus the algorithm must terminate, which yields contradiction. This concludes the proof that a source \mathcal{S} is a match for a query q iff $\text{FindMatch}(I, \aleph, q) \neq \perp$.

Next, we demonstrate that the semantic score of \mathcal{S} is $v = |\text{FindMatch}(I, \aleph, q)|$. If the algorithm returns \perp , then $v = \infty$ by definition, and thus the thesis is proven. Otherwise, according to Definition 7, we need to prove that there exist F and s such that $v = \Delta(\gamma(\varepsilon(I, \aleph), F)) + \Delta(s)$ and that v is minimal among all possible choices of F and s satisfying conditions (c1) and (c2) from Definition 6. By construction of $\Pi_{AD}(I, \mathcal{F} \setminus \mathcal{D}, \aleph^\times)$, Definition 4, and the earlier part of the present theorem, it follows that $\Delta(\gamma(\varepsilon(I, \aleph), F))$ is equal to the number of atoms of A formed by relation *forced*. Similarly, $\Delta(s)$ is equal to the number of atoms of A formed by *split*. Hence, $v = \Delta(\gamma(\varepsilon(I, \aleph), F)) + \Delta(s)$. The minimality of v is demonstrated by contradiction. Let us proceed by cases. Suppose that, when the algorithm terminates at step (4c), $F = \emptyset$ and $s = \aleph^?$. By Definition 4 and Definition 6, v is minimal, which yields contradiction. Suppose, then, that $F \neq \emptyset$ or $s \neq \aleph^?$. Because the values of the two variables are changed only by step 7, it follows that they were set at that step from the values of F' and s' determined by step 5. However, the values of those variables are selected so that $|F'| + \Delta(s')$ is minimal (step 5b). Contradiction.

□

Appendix C Addendum: Figures

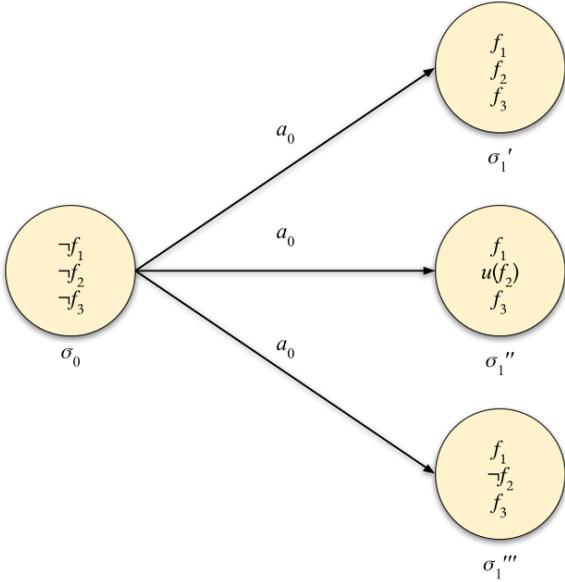


Fig. C 1. Sample transitions

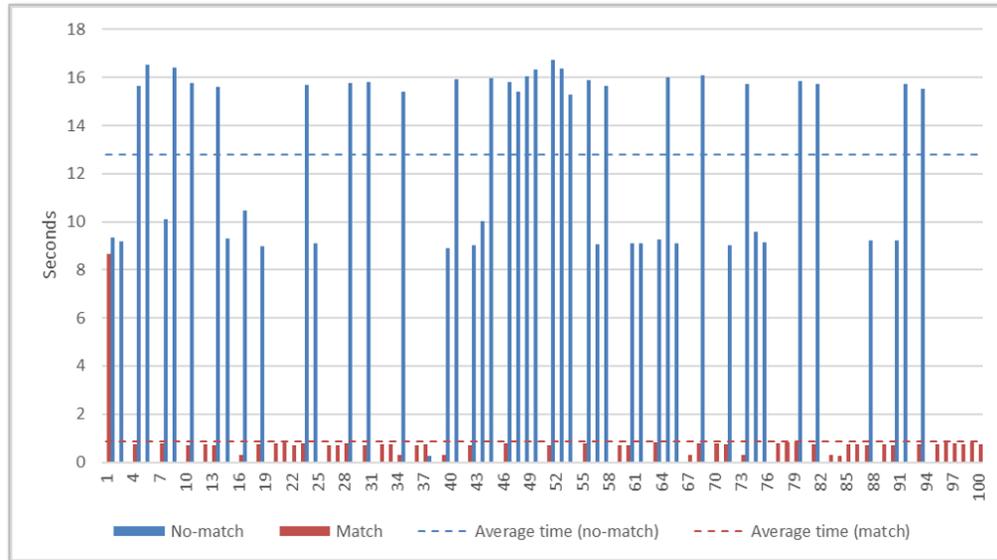


Fig. C2. Sensitivity to problem features

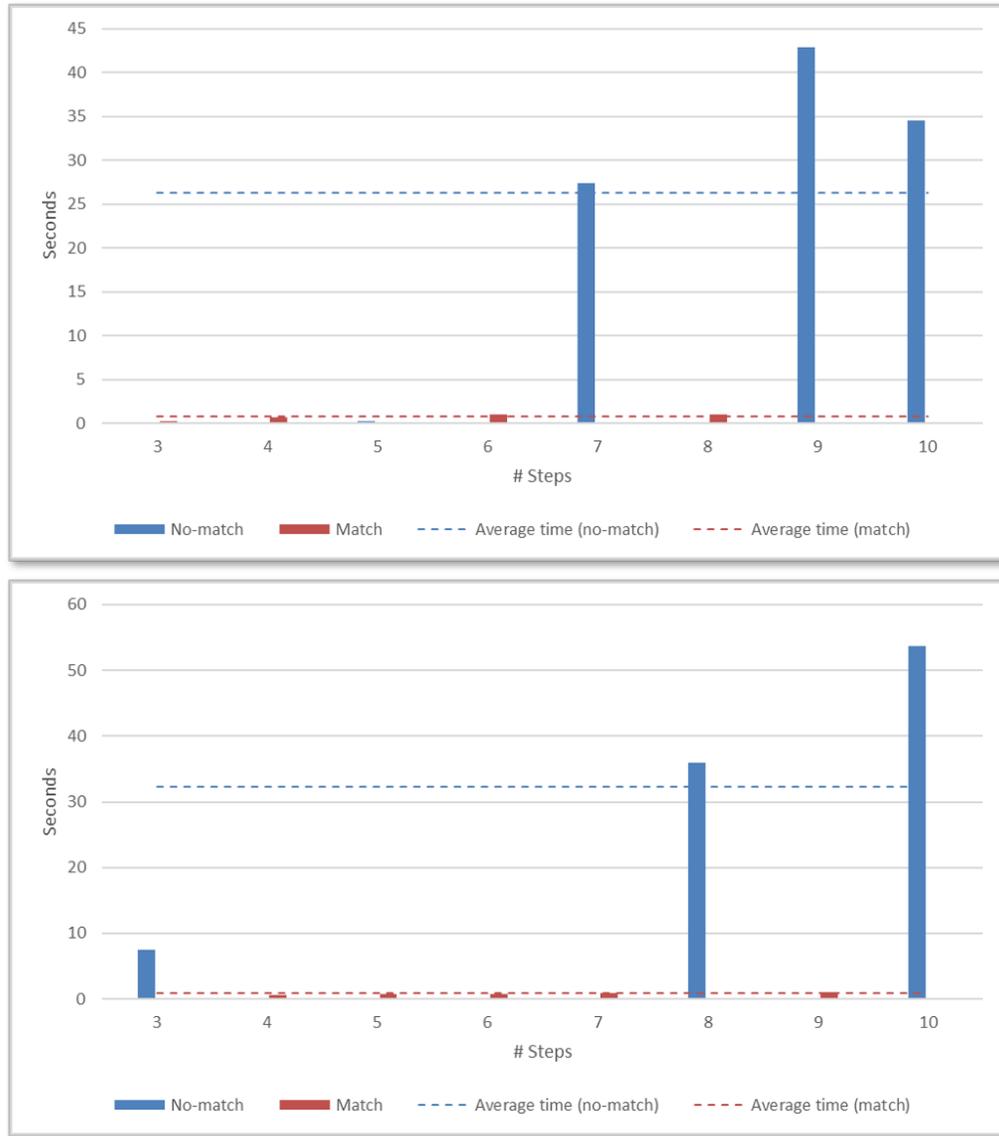


Fig. C 3. Sensitivity to the number of actions, instance set #1 (top) and instance set #2 (bottom)

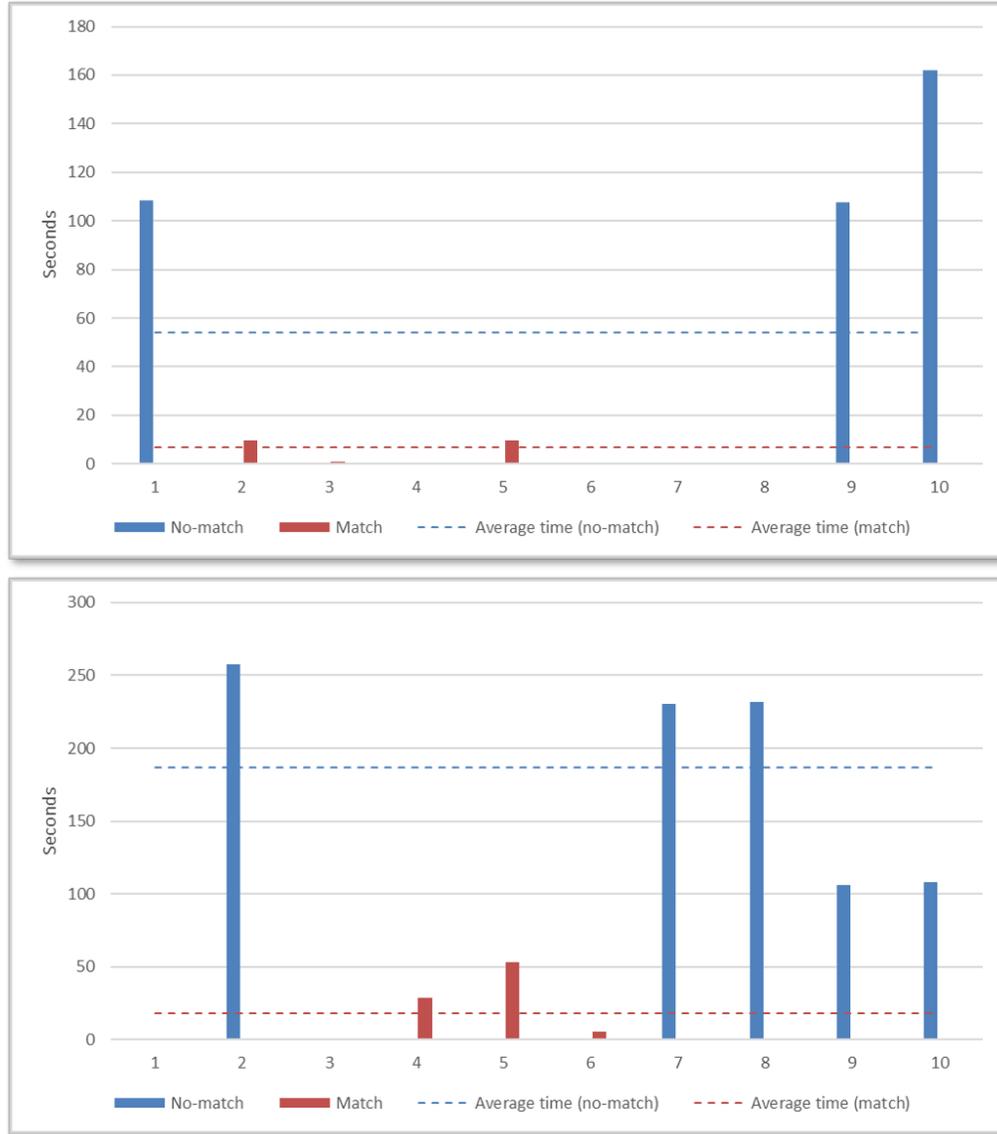


Fig. C 4. Sensitivity to non-determinism, instance set #1 (top) and instance set #2 (bottom)