Online Action Language $o\mathcal{BC}+$

Joseph Babb and Joohyung Lee

School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe, USA
{Joseph.Babb, joolee}@asu.edu

Abstract. We present an online action language called $o\mathcal{BC}+$, which extends action language $\mathcal{BC}+$ to handle external events arriving online. This is done by first extending the concept of online answer set solving to arbitrary propositional formulas, and then defining the semantics of $o\mathcal{BC}+$ based on this extension, similar to the way the offline $\mathcal{BC}+$ is defined. The design of $o\mathcal{BC}+$ ensures that any action description in $o\mathcal{BC}+$ satisfies the syntactic conditions required for the correct computation of online answer set solving, thereby alleviates the user's burden for checking the sophisticated conditions.

1 Introduction

While Answer Set Programming (ASP) is being widely applied to many challenging problems, most ASP applications are limited to offline usages. Continuous grounding and solving in view of possible yet unknown future events, such as the one required for the emerging applications in stream reasoning [1], is one of the main challenges in applying ASP to real-time dynamic systems.

Recently, there emerged the concept of *reactive answer set programming* [2], which is to incrementally ground and compose program slices taking into account external knowledge acquired asynchronously, thereby avoiding multiple unnecessary restarts of the grounding and solving process for each arrival of external inputs. For this, an online ASP program consists of multiple subprograms of different roles, and certain syntactic restrictions originating from the *module theorem* [3] are imposed to ensure the compositionality of their answer sets. The work led to an implementation OCLINGO,[1] which extends ASP grounder GRINGO and ASP solver CLASP in a monolithic way to handle external modules provided at runtime by a controller. However, checking the syntactic requirement for sound execution of online answer set solving is quite a complex task for the user, which significantly limits the usability of online answer set programming.

We address this challenge by introducing an online extension of high level action language $\mathcal{BC}+$ [4], which we call $o\mathcal{BC}+$. $\mathcal{BC}+$ is a recently proposed action language whose semantics is defined in terms of propositional formulas under the stable model semantics. It is shown in [4] that $\mathcal{BC}+$ is expressive enough to embed other action languages, such as $\mathcal{B}$, $\mathcal{C}$, $\mathcal{C}+$ [5] and $\mathcal{BC}$ [6]. Thus $o\mathcal{BC}+$ can be viewed as online extensions of these languages as well.

Since the semantics of $\mathcal{BC}+$ is based on propositional formulas under the stable model semantics, we first generalize the result on online answer set solving to arbitrary

[1] http://www.cs.uni-potsdam.de/wv/oclingo/

propositional formulas, and define $o\mathcal{BC}$+ based on it. We demonstrate that $o\mathcal{BC}$+ provides a structured input language for online answer set solving, and thereby alleviates the user's burden for checking sophisticated conditions imposed on the input programs.

The paper is organized as follows. Section 2 reviews language $\mathcal{BC}$+ from [4] and the module theorem from [7]. Section 3 extends the concept of online answer set solving to propositional theories, based on which Section 4 defines the online extension of $\mathcal{BC}$+, and asserts that the design of the language ensures the syntactic conditions for applying online answer set solving.

2 Preliminaries

2.1 Review: Stable Models of Propositional Formulas

According to [8], stable models of a propositional formula are defined as follows. The *reduct* F^X of a propositional formula F relative to a set X of atoms is the formula obtained from F by replacing every maximal subformula that is not satisfied by X with $\bot$. Set X is called a *stable model* of F if it is a minimal set of atoms satisfying F^X. It is known that propositional logic programs can be identified with propositional formulas under the stable model semantics in the form of conjunctions of implications.

Throughout this paper, we consider propositional formulas whose signature σ consists of atoms of the form $c = v$, [2] where c is called a *constant* and is associated with a finite set $Dom(c)$ of cardinality ≥ 2, called the *domain*, and v is an element of its domain. If the domain of c is $\{\mathbf{f}, \mathbf{t}\}$ then we say that c is *Boolean*, and abbreviate $c = \mathbf{t}$ as c and $c = \mathbf{f}$ as $\sim c$.

2.2 Review: $\mathcal{BC}$+

Syntax Language $\mathcal{BC}$+ includes two kinds of constants, *fluent constants* and *action constants*. Fluent constants are further divided into *regular* and *statically determined*. [3]

A *fluent formula* is a formula such that all constants occurring in it are fluent constants. An *action formula* is a formula that contains at least one action constant and no fluent constants.

A *static law* is an expression of the form

$$\textbf{caused } F \textbf{ if } G \tag{1}$$

where F and G are fluent formulas. An *action dynamic law* is an expression of the form (1) in which F is an action formula and G is a formula. A *fluent dynamic law* is an expression of the form

$$\textbf{caused } F \textbf{ if } G \textbf{ after } H \tag{2}$$

where F and G are fluent formulas and H is a formula, provided that F does not contain statically determined constants. Static laws can be used to talk about causal dependencies between fluents in the same state. Action dynamic laws can be used to

[2] So $c = v$ is an atom in the propositional signature, and not an equality in first-order logic.

[3] Statically determined fluents are fluents whose values are completely determined by fluents in the same state, and not by direct effects of actions [5, Section 5.5].

express causal dependencies between concurrently executed actions. A more common use of action dynamic laws is to express the assumption of an action being "exogenous" (the cause of the action is outside the domain description). Fluent dynamic laws can be used for describing direct effects of actions.

A *causal law* is a static law, an action dynamic law, or a fluent dynamic law. An *action description* is a finite set of causal laws.

The formula F in a causal law (1) or (2) is called the *head*, and G and H are called the *bodies*.

Semantics of $\mathcal{BC}$+ The semantics of $\mathcal{BC}$+ can be understood in terms of a "transition system"—a directed graph whose vertices are states of the world and edges represent transitions between states. For any action description D with a set σ^{fl} of fluent constants and a set σ^{act} of action constants, we define a sequence of propositional formulas $PF_0(D), PF_1(D), \ldots$ so that the stable models of $PF_m(D)$ represent paths of length m in the transition system corresponding to D. The signature of $PF_m(D)$ consists of atoms of the form $i:c=v$ such that

- for each fluent constant c of D, $i \in \{0,\ldots,m\}$ and $v \in Dom(c)$, and
- for each action constant c of D, $i \in \{0,\ldots,m-1\}$ and $v \in Dom(c)$.

By $i:F$ we denote the result of inserting $i:$ in front of every occurrence of every constant in formula F.

For any set $\mathbf{c}$ of symbols from σ^{fl} and σ^{act}, by $UEC_{\mathbf{c}}$ we denote the conjunction of

$$\bigwedge_{v\neq w \mid v,w\in Dom(c)} \neg(c=v \wedge c=w) \quad \wedge \quad \neg\neg \bigvee_{v\in Dom(c)} c=v\,, \tag{3}$$

for all $c \in \mathbf{c}$, which represents the uniqueness and existence of values for the constants in $\mathbf{c}$.

For any atom $c=v$, "choice rule" $\{c=v\}^{\mathrm{ch}}$ stands for $c=v \vee \neg(c=v)$, which, in the presence of (3), means that by default c is mapped to v [9].

The translation $PF_m(D)$ is the conjunction of

$j:F \leftarrow j:G$	for each static law (1) in D
$i:F \leftarrow i:G$	for each action dynamic law (1) in D
$(i+1):F \leftarrow (i+1):G \wedge i:H$	for each fluent dynamic law (2) in D
$\{0:c=v\}^{\mathrm{ch}}$	for each regular fluent c and every $v \in Dom(c)$
$j:UEC_{\sigma^{fl}} \qquad i:UEC_{\sigma^{act}}$	
$(i=0,\ldots,m-1,\ \ j=0,\ldots m)$.	

We identify an interpretation I with the set of atoms that are satisfied by this interpretation. This allows us to represent any interpretation of the signature of $PF_m(D)$ in the form

$$(0:s_0) \cup (0:e_0) \cup (1:s_1) \cup (1:e_1) \cup \cdots \cup (m:s_m)$$

where $s_0,\ldots,s_m$ are interpretations of σ^{fl} and $e_0,\ldots,e_{m-1}$ are interpretations of σ^{act}.

States and transitions are defined in terms of stable models of $PF_0(D)$ and $PF_1(D)$ as follows.

Definition 1 (States and Transitions). *For any action description D of signature σ, a* state *of D is an interpretation s of σ^{fl} such that $0:s$ is a stable model of $PF_0(D)$. A* transition *of D is a triple $\langle s, e, s'\rangle$ where s and s' are interpretations of σ^{fl} and e is an interpretation of σ^{act} such that $0:s \cup 0:e \cup 1:s'$ is a stable model of $PF_1(D)$.*

In view of the uniqueness and existence of value constraints for every state s and every fluent constant c, there exists exactly one v such that $c=v$ belongs to s; this v is considered the value of c in state s.

Given these definitions, we define the transition system $T(D)$ represented by an action description D as follows.

Definition 2 (Transition System). *A transition system $T(D)$ represented by an action description D is a labeled directed graph such that the vertices are the states of D, and the edges are obtained from the transitions of D as follows: for every transition $\langle s, e, s'\rangle$ of D, an edge labeled e goes from s to s'.*

Since the vertices and the edges of a transition system $T(D)$ are identified with the states and the transitions of D, we simply apply the definitions of a state and a transition to transition systems: A *state* of $T(D)$ is a state of D. A *transition* of $T(D)$ is a transition of D.

The stable models of $PF_m(D)$ represent the paths of length m in the transition system represented by D [4, Theorem 2].

2.3 Review: Module Theorem

We review the module theorem from [7] limited to the propositional case.

For any propositional formula F, by $At(F)$ we denote the set of all atoms occurring in F. The *head* atoms of F are defined to be the atoms that has an occurrence in F that is not in the antecedent of any implication (we understand $\neg F$ as an abbreviation of $F \to \bot$). By $Head(F)$ we denote the set of all head atoms of F.

A *module* $\mathbb{F}$ is a triple $(F, \mathcal{I}, \mathcal{O})$, where F is a propositional formula, and $\mathcal{I}$ and $\mathcal{O}$ are disjoint sets of atoms such that $At(F) \subseteq (\mathcal{I} \cup \mathcal{O})$.

Definition 3 (Module Stable Model). *We say that an interpretation I is a* (module) stable model *of a module $\mathbb{F} = (F, \mathcal{I}, \mathcal{O})$ if I is a stable model of $F \wedge \bigwedge_{A\in\mathcal{I}}\{A\}^{\mathrm{ch}}$.*

We refer the reader to [10] for the definition of a dependency graph of a propositional formula F relative to a set A of atoms, which we denote by $\mathrm{DG}[F;\ A]$.

Definition 4 (Joinability of Modules). *Two modules $\mathbb{F}_1 = (F_1 \wedge H,\ \mathcal{I}_1,\ \mathcal{O}_1)$ and $\mathbb{F}_2 = (F_2 \wedge H,\ \mathcal{I}_2,\ \mathcal{O}_2)$ are called* joinable *if*

- $\mathcal{O}_1 \cap \mathcal{O}_2 = \emptyset$,
- *each strongly connected component of* $\mathrm{DG}[F_1 \wedge F_2 \wedge H;\ \mathcal{O}_1 \cup \mathcal{O}_2]$ *is either a subset of $\mathcal{O}_1$ or a subset of $\mathcal{O}_2$,*
- *$Head(F_1) \cap \mathcal{O}_2 = \emptyset$, and $Head(F_2) \cap \mathcal{O}_1 = \emptyset$.*

Definition 5 (Join of Modules). *For any modules $\mathbb{F}_1 = (F_1 \wedge H,\ \mathcal{I}_1,\ \mathcal{O}_1)$ and $\mathbb{F}_2 = (F_2 \wedge H,\ \mathcal{I}_2,\ \mathcal{O}_2)$ that are joinable, the* join *of $\mathbb{F}_1$ and $\mathbb{F}_2$, denoted by $\mathbb{F}_1 \sqcup \mathbb{F}_2$, is defined to be the module $(F_1 \wedge F_2 \wedge H,\ (\mathcal{I}_1 \cup \mathcal{I}_2) \setminus (\mathcal{O}_1 \cup \mathcal{O}_2),\ \mathcal{O}_1 \cup \mathcal{O}_2)$.*

Given sets of atoms I_1, I_2, I_3, we say that I_1 and I_2 are I_3-compatible if $I_1 \cap I_3 = I_2 \cap I_3$.

Theorem 1 (Module Theorem [7]) *Let $\mathbb{F}_1 = (F_1, \mathcal{I}_1, \mathcal{O}_1)$ and $\mathbb{F}_2 = (F_2, \mathcal{I}_2, \mathcal{O}_2)$ be modules that are joinable, and let I_i ($i = 1, 2$) be a subset of $(\mathcal{I}_i \cup \mathcal{O}_i)$ such that I_1 and I_2 are $(\mathcal{I}_1 \cup \mathcal{O}_1) \cap (\mathcal{I}_2 \cup \mathcal{O}_2)$-compatible. Then $I_1 \cup I_2$ is a stable model of $\mathbb{F}_1 \sqcup \mathbb{F}_2$ iff I_1 is a stable model of $\mathbb{F}_1$ and I_2 is a stable model of $\mathbb{F}_2$.*

3 Online Propositional Theories

We generalize the concept of online answer set solving to arbitrary propositional formulas as follows. This section inevitably has many notions, all of which are generalized from those in [2]. The generalization will be used in the next section in order to extend $\mathcal{BC}$+ to the online setting.

A *step-parametrized formula* $F[t]$ is a propositional formula which may contain *step-parameterized atoms* of the form $g(t) : a$, where t is a variable for nonnegative integers denoting a *step counter*, and $g(t)$ is some meta-level nonnegative integer valued arithmetic function whose only free variable is t. Given such a formula $F[t]$ and a nonnegative integer k, the *step-instantiated formula* $F[t/k]$ (or simply $F[k]$) is defined to be the propositional formula which is obtained from $F[t]$ by replacing every occurrence of every step-parametrized atom $g(t) : a$ with a standard atom $v : a$, where v is the value of $g(k)$. (Thus, $v : a$ is assumed to be in the underlying propositional signature.)

We define an *incremental theory* to be a triple $\langle B, P[t], Q[t] \rangle$ such that B is a propositional formula, and $P[t]$, $Q[t]$ are step-parametrized formulas. Informally, B is the *base component*, which describes static knowledge; $P[t]$ is the *cumulative component*, which contains information regarding every step that should be accumulated during execution; $Q[t]$ is the *volatile component*, which contains constraints or other information regarding the final step.

By an *online progression* $\langle E, F \rangle$ we denote some sequence of pairs of step-instantiated formulas $(E_i[e_i], F_i[f_i])$ for $i \geq 1$ with associated nonnegative integers e_i, f_i such that $e_i \leq f_i$. Intuitively, each $E_i[e_i]$ and $F_i[f_i]$ corresponds to stable and volatile knowledge acquired during execution, respectively. For each $(E_i[e_i], F_i[f_i])$, e_i and f_i denote the step for which they are relevant allowing knowledge to be acquired out of order. For example, $E_4[3]$ is the fourth piece of online input and contains information relevant to step 3.

Given an incremental theory $\langle B, P[t], Q[t] \rangle$, an online progression $\langle E, F \rangle$, and nonnegative integers j, k such that $e_1, \ldots, e_j, f_j \leq k$, the *incremental components* are

$$\{B, P[t/1], P[t/2], \ldots, P[t/k], Q[t/k], E_1[e_1], E_2[e_2], \ldots, E_j[e_j], F_j[f_j]\}. \quad (4)$$

As in [2], we define the *k-expanded propositional formula* $R_{j,k}$ of $\langle B, P[t], Q[t] \rangle$ w.r.t. $\langle E, F \rangle$ to be the conjunction of all formulas in (4).

Generalizing the notion of the simplification in [2], given a propositional formula F, we define the *simplification* of F onto a set A of atoms (denoted $Simplify(F, A)$) to be the formula obtained from F by replacing all occurrences of atoms p in F such that

$p \notin (Head(F) \cup A)$ with $\bot$ and performing the following syntactic transformations recursively until no further transformations are possible:[4]

$$\begin{array}{llll} \neg\bot \ \mapsto\ \top & \neg\top \ \mapsto\ \bot & & \\ \bot \wedge F \ \mapsto\ \bot & F \wedge \bot \ \mapsto\ \bot & \top \wedge F \ \mapsto\ F & F \wedge \top \ \mapsto\ F \\ \bot \vee F \ \mapsto\ F & F \vee \bot \ \mapsto\ F & \top \vee F \ \mapsto\ \top & F \vee \top \ \mapsto\ \top \\ \bot \rightarrow F \ \mapsto\ \top & F \rightarrow \top \ \mapsto\ \top & \top \rightarrow F \ \mapsto\ F & \end{array}$$

We define the *modular instantiation* of F with respect to A, denoted $PM(F, A)$, to be the module $(Simplify(F, A),\ A,\ At(Simplify(F, A))\setminus A)$.[5]

The idea of the simplification is to reduce the size of the input formulas by exploiting the fact that some atoms are known not to belong to any stable model. However, unlike the offline solving, the values of external atoms in the online incremental computation are unknown at the time of simplifying the current module containing them. Thus, following [2], we associate each formula F in (4) with some designated set of external atoms $I(F)$ such that $Head(F) \cap I(F) = \emptyset$. Such atoms represent possible external inputs that may be introduced later by an online progression, and thus should be exempted from the current program simplification.

Given a module $\mathbb{F} = \langle F, \mathcal{I}, \mathcal{O}\rangle$, $Out(\mathbb{F})$ refers to $\mathcal{O}$.

Definition 6 (Modular Incremental Theories and Online Progression). *We say that an incremental theory $\langle B, P[t], Q[t]\rangle$ and an online progression $\langle E, F\rangle$ are* modular *if the following modules are well-defined for any nonnegative integers j, k such that $e_1, \ldots, e_j, f_j \leq k$.*[6]

$$\begin{array}{lll} \mathbb{P}_0 = PM(B,\ I(B)), & \mathbb{E}_0 = \mathbb{F}_0 = \langle \top, \emptyset, \emptyset\rangle, & \\ \mathbb{P}_i = \mathbb{P}_{i-1} \sqcup PM(P[t/i],\ Out(\mathbb{P}_{i-1}) \cup I(P[t/i])), & & (i = 1, \ldots, k) \\ \mathbb{Q}_i = PM(Q[t/i],\ Out(\mathbb{P}_i) \cup I(Q[t/i])), & & (i = 0, \ldots, k) \\ \mathbb{E}_i = \mathbb{E}_{i-1} \sqcup PM(E_i[e_i],\ Out(\mathbb{P}_{e_i}) \cup Out(\mathbb{E}_{i-1}) \cup I(E_i[e_i])) & & (i = 1, \ldots, j) \\ \mathbb{F}_i = PM(F_i[f_i],\ Out(\mathbb{P}_{f_i}) \cup Out(\mathbb{E}_i) \cup I(F_i[f_i])) & & (i = 1, \ldots, j) \\ \mathbb{R}_{j,k} = \mathbb{P}_k \sqcup \mathbb{Q}_k \sqcup \mathbb{E}_j \sqcup \mathbb{F}_j. & & \end{array}$$

We refer to $\mathbb{R}_{j,k}$ as the *incremental composition* of the incremental theory $\langle B, P[t], Q[t]\rangle$ w.r.t. the online progression $\langle E, F\rangle$. Unlike the k-expanded propositional formula $R_{j,k}$, each component in the incrementally composed module $\mathbb{R}_{j,k}$ is simplified before being joined.

Given an incremental theory $\langle B, P[t], Q[t]\rangle$ and an online progression $\langle E, F\rangle$, we assume the precedence relation $\prec^*$ on the set

$$\begin{array}{c} \{B, P[t/1], P[t/2], \ldots, P[t/k], E_1[e_1], E_2[e_2], \ldots, E_j[e_j], \\ Q[t/0], Q[t/1], \ldots, Q[t/k], F_1[f_1], F_2[f_2], \ldots, F_j[f_j]\} \end{array} \tag{5}$$

[4] In [2], this process stops only at the second iteration.

[5] In practice when F is non-ground, we assume F is grounded first by substituting every variable with every element in the Herbrand universe.

[6] For notational simplicity, we define $E_0[e_0]$ and $F_0[f_0]$ to be $\top$, e_0, f_0 to be 0, and $I(E_0[e_0])$ and $I(F_0[f_0])$ to be $\emptyset$.

as the transitive closure of the following relation $\prec$:

$$\begin{array}{lll} B \prec P[t/1] \prec \cdots \prec P[t/k], & B \prec Q[t/0], \quad P[t/i] \prec Q[t/i] & (i \geq 1) \\ E_1[e_1] \prec \cdots \prec E_j[e_j], & E_i[e_i] \prec F_i[f_i] & (i \geq 1) \\ P[t/e_i] \prec E_i[e_i], & P[t/f_i] \prec F_i[f_i] & (i \geq 1). \end{array}$$

Additionally, we say that two formulas F and G in (5) *coexist* if they belong to (4) for some nonnegative integers j, k. Intuitively, F and G coexist if they are eventually composed together into some $\mathbb{R}_{j,k}$. For example, $P[t/1]$ and $Q[t/3]$ coexist as they are both present in $\mathbb{R}_{0,3}$, whereas $Q[t/1]$ and $Q[t/3]$ do not. In Figure 1 the shaded blocks denote coexisting formulas for $j = 1, k = 2$.

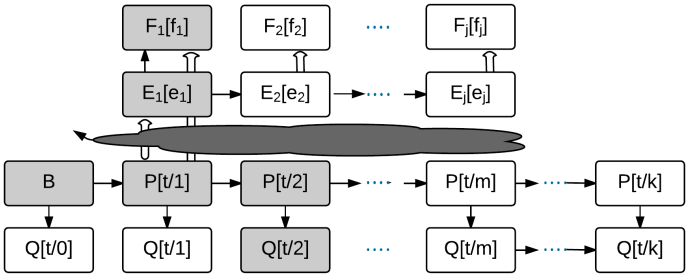

Fig. 1. Precedence Graph of Component Formulas

Definition 7. *We say that an incremental theory $\langle B, P[t], Q[t]\rangle$ and an online progression $\langle E, F\rangle$ are* mutually revisable *if, for any distinct coexisting formulas G and H in* (5)*, we have that $G \prec^* H$ whenever $Head(G) \cap (At(H) \setminus I(H)) \neq \emptyset$.*

Theorem 2 (Correctness of Incremental Composition) *Given an incremental theory $\langle B, P[t], Q[t]\rangle$ and an online progression $\langle E, F\rangle$ which are modular and mutually revisable, and nonnegative integers j, k such that $e_1, \ldots, e_j, f_j \leq k$, let $R_{j,k}$ be the k-expanded propositional formula w.r.t. $\langle E, F\rangle$, and let $\mathbb{R}_{j,k} = (H, \mathcal{I}, \mathcal{O})$ be the incremental composition of $\langle B, P[t], Q[t]\rangle$ w.r.t. $\langle E, F\rangle$. Then the stable models of $R_{j,k}$ coincide with the stable models of H, the formula of $\mathbb{R}_{j,k}$.*

It turns out that in the event that all explicit inputs (all $I(F)$ for $F \in$ (4)) are empty, i.e., in the offline case, mutual revisability is a stronger condition than modularity. This means that in the offline case it is sufficient to just check that an incremental theory is mutually revisable.

Using Theorem 2, it is possible to incrementally ground, simplify, and solve a traditional ASP incremental theory in order to find the minimum k such that $R_{j,k}$ has an answer set without repeating previous work performed. In practice, this allows for a significant speedup when performing an iterative deepening search, such as when searching for a minimum length plan to accomplish a goal. In addition, the system is able to account for specific forms of online input in an equally efficient manner by allowing external information to be asserted in the online progression during execution.

Example 1 (Online ASP Solving) *Given an incremental theory*

$$\langle B, P[t], Q[t]\rangle = \langle \top,\ \neg((t-1):q) \land \neg((t-1):p) \to t:p,\ \neg(t:p) \to \bot\rangle$$

such that $I(B) = \emptyset$, $I(P[t/i]) = \{(i-1):q\}$, *and* $I(Q[t/i]) = \emptyset$ *and an online progression* $\langle E, F\rangle$.

Initially, $\mathbb{R}_{0,0}$ *is constructed such that*

$$\begin{aligned}
\mathbb{P}_0 &= \langle \top, \emptyset, \emptyset\rangle,\\
\mathbb{Q}_0 &= PM(Q[t/0], \mathit{Out}(\mathbb{P}_0) \cup I(Q[t/0])) = \langle \top \to \bot, \emptyset, \emptyset\rangle,\\
\mathbb{R}_{0,0} &= \mathbb{P}_0 \sqcup \mathbb{Q}_0 = \langle \top \to \bot, \emptyset, \emptyset\rangle.
\end{aligned}$$

Clearly, $\top \to \bot$ *has no stable models. As a result,* $\mathbb{R}_{0,1}$ *is attempted as follows:*

$$\begin{aligned}
\mathbb{P}_1 &= \mathbb{P}_0 \sqcup PM(P[t/1], \mathit{Out}(\mathbb{P}_0) \cup I(P[t/1])) = \mathbb{P}_0 \sqcup \langle \neg(0:q) \to 1:p, \{0:q\}, \{1:p\}\rangle\\
&= \langle \neg(0:q) \to 1:p, \{0:q\}, \{1:p\}\rangle,\\
\mathbb{Q}_1 &= PM(Q[t/1], \mathit{Out}(\mathbb{P}_1) \cup I(Q[t/1])) = \langle \neg(1:p) \to \bot, \{1:p\}, \emptyset\rangle,\\
\mathbb{R}_{0,1} &= \mathbb{P}_1 \sqcup \mathbb{Q}_1 = \langle (\neg(0:q) \to 1:p) \land (\neg(1:p) \to \bot), \{0:q\}, \{1:p\}\rangle.
\end{aligned}$$

Solving is then halted as the formula in $\mathbb{R}_{0,1}$ *has one stable model* $\{1:p\}$. *However, with the arrival of the external event* $E_1[0] = 0:q$, *and* $F_1[0] = \top$ *such that* $I(E_1[0]) = I(F_1[0]) = \emptyset$, *we then must consider the construction of* $\mathbb{R}_{1,k}$, *rather than* $\mathbb{R}_{0,k}$. $\mathbb{R}_{1,1}$ *is constructed such that*

$$\begin{aligned}
\mathbb{E}_1 &= \langle \top, \emptyset, \emptyset\rangle \sqcup PM(E_1[e_1], \mathit{Out}(\mathbb{P}_{e_1}) \cup \mathit{Out}(\mathbb{E}_0) \cup I(E_1[e_1])) && (e_1 \text{ is } 0)\\
&= \langle \top, \emptyset, \emptyset\rangle \sqcup \langle 0:q, \emptyset, \{0:q\}\rangle = \langle 0:q, \emptyset, \{0:q\}\rangle,\\
\mathbb{F}_1 &= PM(F_1[f_1], \mathit{Out}(\mathbb{P}_{f_1}) \cup \mathit{Out}(\mathbb{E}_1) \cup I(F_1[f_1])) && (f_1 \text{ is } 0)\\
&= \langle \top, \{0:q\}, \emptyset\rangle,\\
\mathbb{R}_{1,1} &= \mathbb{P}_1 \sqcup \mathbb{Q}_1 \sqcup \mathbb{E}_1 \sqcup \mathbb{F}_1 = \langle (\neg(0:q) \to 1:p) \land 0:q \land (\neg(1:p) \to \bot),\ \emptyset,\ \{0:q, 1:p\}\rangle.
\end{aligned}$$

Once again, the formula of $\mathbb{R}_{1,1}$ *has no stable models, so the search is deepened to* $\mathbb{R}_{1,2}$ *as follows:*

$$\begin{aligned}
\mathbb{P}_2 &= \mathbb{P}_1 \sqcup PM(P[t/2], \mathit{Out}(\mathbb{P}_1) \cup I(P[t/2]))\\
&= \mathbb{P}_1 \sqcup \langle \neg(1:q) \land \neg(1:p) \to 2:p, \{1:q, 1:p\}, \{2:p\}\rangle\\
&= \langle (\neg(0:q) \to 1:p) \land (\neg(1:q) \land \neg(1:p) \to 2:p), \{0:q, 1:q\}, \{1:p, 2:p\}\rangle,\\
\mathbb{Q}_2 &= PM(Q[t/2], \mathit{Out}(\mathbb{P}_2) \cup I(Q[t/2])) = \langle \neg(2:p) \to \bot, \{1:p, 2:p\}, \emptyset\rangle,\\
\mathbb{R}_{1,2} &= \mathbb{P}_2 \sqcup \mathbb{Q}_2 \sqcup \mathbb{E}_1 \sqcup \mathbb{F}_1\\
&= \langle (\neg(0:q) \to 1:p) \land (\neg(1:q) \land \neg(1:p) \to 2:p) \land 0:q \land (\neg(2:p) \to \bot),\\
&\qquad\qquad \{1:q\},\ \{0:q, 1:p, 2:p\}\rangle.
\end{aligned}$$

The formula of $\mathbb{R}_{1,2}$ *has a single stable model* $\{0:q, 2:p\}$.

4 Online Execution of $o\mathcal{BC}+$

Based on the concept of online propositional theories in the previous section, we define an online extension of $\mathcal{BC}+$, which provides a structured input language for online answer set solving that ensures the syntactic conditions of modularity and mutual revisability.

4.1 Syntax

The signature hierarchy of $o\mathcal{BC}$+ is extended from that of (offline) $\mathcal{BC}$+ by adding new sets of symbols called *external fluent constants* (denoted σ^{ef}) and *external action constants* (denoted σ^{ea}) such that $\sigma^{ef} \subseteq \sigma^{fl}$ and $\sigma^{ea} \subseteq \sigma^{act}$. We assume that the domain of each external fluent and action constant contains a special element u, which represents an *unknown* value. The syntax of causal laws is defined the same as in Section 2.2 except that external constants are allowed in the bodies but not in the heads.

An *observation* is an expression of the form

observed $c{=}v$ **at** m (6)

where $c{=}v$ is an atom such that c is an external constant, v is a value other than u, and m is a nonnegative integer. An *observational constraint* is an expression of the form

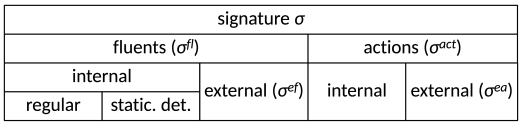

Fig. 2. Hierarchy of $o\mathcal{BC}+$ signature

constraint F **at** m (7)

where F is a propositional formula containing no external constants and m is a nonnegative integer. We say that an observation (6) or observational constraint (7) is *dynamic* if it contains some action constant, otherwise we say it is *static*.

An *observation stream*, denoted $\mathcal{O}_{n,\widehat{m}}$, is a list $O_1, \ldots O_n$ such that

- for each $1 \leq i \leq n$, O_i is a finite set of observations (6) and observational constraints (7), and m_i is the maximum of each m among the static observations and constraints and $m+1$ among the dynamic observations and constraints;
- $\widehat{m}$ is the maximum of each m_i $(1 \leq i \leq n)$;
- for each external constant c and each m in $\{1, \ldots, \widehat{m}\}$, there is at most one observation (6) in $O_1 \cup \cdots \cup O_n$.

4.2 Semantics

Since any future external constants can take any values arbitrarily, the transition system in the presence of external constants can be defined straightforwardly by assigning arbitrary values to the external constants. That is, given an $o\mathcal{BC}+$ description and length m, we extend the propositional formula $PF_m(D)$ in Section 2.2 by adding the formulas $\{i\,{:}\,c{=}v\}^{\mathrm{ch}}$ for every external constant c, every $v \in Dom(c)$, and every $i \in \{0, \ldots, m\}$ if c is a fluent constant, and $i \in \{0, \ldots, m{-}1\}$ if c is an action constant.

On the other hand, it is more meaningful to assume that the external input is "abnormal" to the system dynamics, and we want to "minimize" their effects. In other words, rather than arbitrary histories, we are interested in histories which are "normal" with respect to $\mathcal{O}_{n,\widehat{m}}$. Intuitively, in a normal history, the external constants are mapped to an unknown value unless the external observation asserts otherwise.

Formally, a history $\mathcal{H}_k$ of a transition system of length k is a sequence $\langle s_0, e_0, s_1, \ldots, e_{k-1}, s_k\rangle$ such that each $\langle s_i, e_i, s_{i+1}\rangle$ $(0 \leq i \leq k-1)$ is a transition. We say that $\mathcal{H}_k$ satisfies $i : F$ where F is a fluent formula (action formula, respectively) if $s_i \models F$ ($e_i \models F$, respectively). Given an observation stream $\mathcal{O}_{n,\widehat{m}}$ and history $\mathcal{H}_k$ such that $k \geq \widehat{m}$, we say that $\mathcal{H}_k$ *observes* $\mathcal{O}_{n,\widehat{m}}$ if,

- for each observation (6) in $\mathcal{O}_{n,\widehat{m}}$, history $\mathcal{H}_k$ satisfies $m : c = v$, and
- for each observational constraint (7) in $\mathcal{O}_{n,\widehat{m}}$, history $\mathcal{H}_k$ satisfies $m : F$.

We say that $\mathcal{H}_k$ is *normal* with respect to $\mathcal{O}_{n,\widehat{m}}$, if it observes $\mathcal{O}_{n,\widehat{m}}$, and, for each external fluent constant (action constant, respectively) c and each $i \in \{0, \ldots, k\}$ ($i \in \{0, \ldots, k-1\}$, respectively), $\mathcal{H}_k$ satisfies $i : c = \mathtt{u}$ when there is no observation (6) in $\mathcal{O}_{n,\widehat{m}}$ such that $m = i$.

Intuitively, observations are non-monotonic observations the agent has made regarding the defined external actions and fluents. Meanwhile, the observational constraints serve to further limit past histories according to what the agent knows, such as what actions the agent has executed.

Named Sets:	Value:
Status	{on, off}
Boolean	{t, f}
ExtBoolean	{t, f, u}

Notation: s ranges over elements in *Status*; v ranges over elements in *Boolean*.

Constants:	Type:	Domain:
Sw	regular fluent	*Status*
Light	statically determined fluent	*Status*
Flip	action	*Boolean*
Fault	regular fluent	*ExtBoolean*
ExtFault	external fluent	*ExtBoolean*
ReplaceBulb	action	*Boolean*

Causal laws:

inertial *Sw*	**inertial** *Fault* **after** *ReplaceBulb* = f
exogenous *Flip*	**exogenous** *ReplaceBulb*
Flip **causes** *Sw* = on **if** *Sw* = off	**nonexecutable** *ReplaceBulb* **if** *Flip* = t
Flip **causes** *Sw* = off **if** *Sw* = on	**caused** *Fault* = v **if** *ExtFault* = v
default *Light* = s **if** *Sw* = s	**caused** *Light* = off **if** *Fault* = t
	default *Fault* = u **after** *ReplaceBulb*

Fig. 3. Online Faulty Switch Elaboration in $o\mathcal{BC}+$.

Example 2 *Consider a light switch problem where the light bulb may be burnt out. In the event this is the case, the light will not turn on until the bulb is replaced. This problem can be formalized in $o\mathcal{BC}+$ as shown in Figure 3.*[7] *Intuitively, Fault is the agent's internal model of whether the light is burnt out, while ExtFault represents the agent's external observations.*

Normally, Fault is governed by inertia. However, in the event the agent gains additional information (i.e. observes whether there has been a fault) Fault is updated to reflect this. Performing ReplaceBulb will then reset the agent's internal model and the agent once again assumes that the fault has been fixed.

[7] It uses several abbreviations of causal laws as defined in [4].

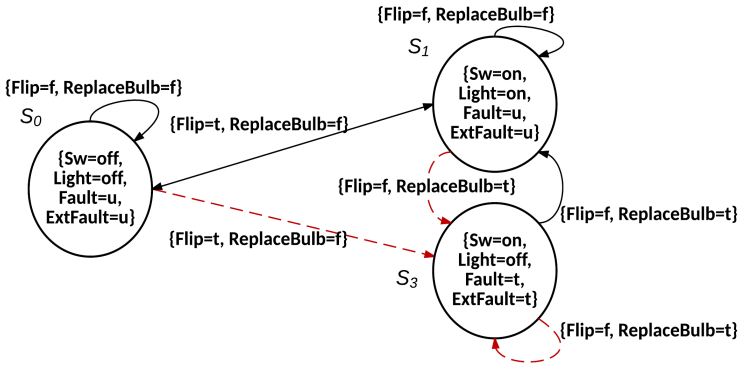

Fig. 4. A Partial Transition System of $\mathcal{D}^{o\mathcal{BC}+}_{\text{switch}}$

Consider the transition system corresponding to the toggle switch elaboration. The minimum length history from

$$\mathcal{S}_0 = \{Switch = \texttt{off},\ Light = \texttt{off},\ Fault = \texttt{u},\ ExtFault = \texttt{u}\}$$

to a state $\mathcal{S}$ such that $\mathcal{S} \models Light = \texttt{on}$ are $\langle \mathcal{S}_0, \mathcal{E}_0, \mathcal{S}_1 \rangle$, and $\langle \mathcal{S}_0, \mathcal{E}_0, \mathcal{S}_2 \rangle$ where

$$\begin{array}{l} \mathcal{E}_0 = \{Flip = \texttt{t},\ ReplaceBulb = \texttt{f}\}, \\ \mathcal{S}_1 = \{Switch = \texttt{on},\ Light = \texttt{on},\ Fault = \texttt{u},\ ExtFault = \texttt{u}\}, \textit{and} \\ \mathcal{S}_2 = \{Switch = \texttt{on},\ Light = \texttt{on},\ Fault = \texttt{f},\ ExtFault = \texttt{f}\}. \end{array}$$

Intuitively, the difference between $\mathcal{S}_1$ and $\mathcal{S}_2$ is that in $\mathcal{S}_1$ the agent has no knowledge as to whether a fault has occurred (i.e. the bulb has burnt out) whereas in $\mathcal{S}_2$ the agent knows that the light is fine. Of the two, only $\langle \mathcal{S}_0, \mathcal{E}_0, \mathcal{S}_1 \rangle$ is normal with respect to the online progression $\mathcal{O}_{0,0} = [\,]$.

If, following the execution of Flip, the agent observes that a fault did occur, the knowledge can be added to the online progression producing

$$\mathcal{O}_{1,1} = [\{\textbf{observed}\ ExtFault = \texttt{t}\ \textbf{at}\ 1,\ \textbf{constraint}\ Flip = \texttt{t}\ \textbf{at}\ 0\}].$$

(The addition of the constraint enforces that the agent has executed $Flip = \texttt{t}$ and prevents that action from being revised.) The new minimum length history from $\mathcal{S}_0$ to a state $\mathcal{S}$ such that $\mathcal{S} \models Light = \texttt{on}$ and is normal w.r.t. $\mathcal{O}_{1,1}$ is $\langle \mathcal{S}_0, \mathcal{E}_0, \mathcal{S}_3, \mathcal{E}_1, \mathcal{S}_1 \rangle$ where

$$\begin{array}{l} \mathcal{S}_3 = \{Switch = \texttt{on}, Light = \texttt{off}, Fault = \texttt{t}, ExtFault = \texttt{t}\},\ \textit{and} \\ \mathcal{E}_1 = \{Flip = \texttt{f}, ReplaceBulb = \texttt{t}\}. \end{array}$$

This history essentially prescribes that the agent should replace the light bulb in order to attempt to fix the fault.

A partial specification of the transition system is shown in Figure 4. The dashed edges depend on the assertion of an external constant and are not considered for transitions in normal histories.

Given an $o\mathcal{BC}$+ action description D, an observation stream $\mathcal{O}_{n,\widehat{m}}$, and some incrementally parametrized formula $\mathcal{Q}[t]$, we define the corresponding incremental theory $\langle B, P[t], Q[t]\rangle^{D,\mathcal{Q}[t]}$ and the online progression $\langle E, F\rangle^{\mathcal{O}_{n,\widehat{m}}}$ as follows.

$$B = \bigwedge \begin{cases} 0\!:\!F \leftarrow 0\!:\!G & \text{for each static law (1) in } D \\ 0\!:\!\{f\!=\!v\}^{\mathrm{ch}} & \text{for each regular fluent } f \text{ and each } v \in \mathit{Dom}(f) \\ 0\!:\!\{f\!=\!\mathtt{u}\}^{\mathrm{ch}} & \text{for each external fluent } f \\ 0\!:\!\mathit{UEC}_{\sigma^{fl}} & \end{cases}$$

$$P[t] = \bigwedge \begin{cases} t\!:\!F \leftarrow t\!:\!G & \text{for each static law (1) in } D \\ (t\!-\!1)\!:\!F \leftarrow (t\!-\!1)\!:\!G & \text{for each action dynamic law (1) in } D \\ t\!:\!F \leftarrow t\!:\!G \wedge (t\!-\!1)\!:\!H & \text{for each fluent dynamic law (2) in } D \\ t\!:\!\{f\!=\!\mathtt{u}\}^{\mathrm{ch}} & \text{for each external fluent } f \\ (t\!-\!1)\!:\!\{a\!=\!\mathtt{u}\}^{\mathrm{ch}} & \text{for each external action } a \\ t\!:\!\mathit{UEC}_{\sigma^{fl}} & \\ (t\!-\!1)\!:\!\mathit{UEC}_{\sigma^{act}} & \end{cases}$$

$$Q[t] = \neg\neg\mathcal{Q}[t]$$

$$E_i[m_i] = \bigwedge \begin{cases} m_i\!:\!c\!=\!v & \text{for each observation (6)} \in O_i \\ \neg\neg m_i\!:\!F & \text{for each observational constraint (7)} \in O_i \end{cases}$$

$$F_i[m_i] = \top$$

The stable models of their incremental composition represent histories that are normal w.r.t. the observation.

Given a $o\mathcal{BC}$+ signature σ we define $\mathit{At}(\sigma)$ to be the set of atoms $c\!=\!v$ where $c \in \sigma$ and $v \in \mathit{Dom}(c)$. Furthermore, we define $\mathit{At}_{\mathtt{u}}(\sigma)$ to be the set of all such atoms such that $v \neq \mathtt{u}$.

We define the sets of explicit external inputs as follows:

- $I(B) = \mathit{At}_{\mathtt{u}}(0\!:\!\sigma^{ef})$,
- $I(P[t/i]) = \mathit{At}_{\mathtt{u}}(i\!:\!\sigma^{ef} \cup (i\!-\!1)\!:\!\sigma^{ea})$,
- $I(Q[t/i]) = \mathit{At}_{\mathtt{u}}(\bigcup_{0\leq j<i}(j\!:\!\sigma^{ef} \cup j\!:\!\sigma^{ea}) \cup i\!:\!\sigma^{ef})$, and
- $I(E_i) = I(F_i) = \emptyset$.

The following proposition asserts that the translation of an $o\mathcal{BC}$+ description into propositional formulas ensures modularity and mutual revisability.

Theorem 3 (Modular and Mutually Revisable Construction) *Given an $o\mathcal{BC}+$ action description D and an observation stream $\mathcal{O}_{n,\widehat{m}}$, and a step-parameterized formula $\mathcal{Q}[t]$, the corresponding incremental theory $\langle B, P[t], Q[t]\rangle^{D,\mathcal{Q}[t]}$ and the corresponding online progression $\langle E, F\rangle^{\mathcal{O}_{n,\widehat{m}}}$ are modular and mutually revisable.*

The next theorem asserts that the stable models of the incremental assembly represents the histories in the transition system that are normal with respect to the online stream.

Theorem 4 (Correctness of Incremental Assembly) *Given an $o\mathcal{BC}+$ action description D, an observation stream $\mathcal{O}_{n,\widehat{m}}$, a step-parameterized $\mathcal{Q}[t]$, and some $k \geq \widehat{m}$, let $\mathbb{R}_{\widehat{m},k} = \langle H, \mathcal{I}, \mathcal{O}\rangle$ be the incremental composition of $\langle B, P[t], Q[t]\rangle^{D,\mathcal{Q}[t]}$ w.r.t. $\langle E, F\rangle^{\mathcal{O}_{n,\widehat{m}}}$. The stable models of H represents the histories of length k in the transition system described by D which (i) observe $\mathcal{O}_{n,\widehat{m}}$, (ii) are normal with respect to $\mathcal{O}_{n,\widehat{m}}$, and (iii) satisfy $\mathcal{Q}[t/k]$.*

5 Conclusion

We extended the concept of online answer set solving to propositional formulas under the stable model semantics, and based on this, designed a high level online action language $o\mathcal{BC}$+, whose structure ensures the syntactic conditions that are required for the correctness of online answer set solving.

Another high level language for OCLINGO ensuring the modularity condition of OCLINGO is Online Agent Logic Programming language from [11]. However, this is based on Agent Logic Programs, instead of action languages, and lacks negation.

$o\mathcal{BC}$+ is implemented in Version 3 of CPLUS2ASP. In addition to the static and the incremental mode already available in Version 2, which invoke CLINGO v3.0.5 and ICLINGO v3.0.5, respectively, newly introduced is the reactive mode, which invokes OCLINGO v3.0.92. The "reactive bridge" is a new software component, and acts as an intermediary between OCLINGO and a user-provided agent controller system. It allows the agent controller system to provide an $o\mathcal{BC}$+ observation stream during execution and receive updated solutions in the form of transition system histories. We refer the reader to the system homepage (http://reasoning.eas.asu.edu/cplus2asp) for more details and experiment results.

Acknowledgements: We are grateful to Michael Bartholomew, Yi Wang, and the anonymous referees for their useful comments on the draft. This work was partially supported by the National Science Foundation under Grant IIS-1319794 and South Korea IT R&D program MKE/KIAT 2010-TD-300404-001.

References

1. Valle, E.D., Ceri, S., van Harmelen, F., Fensel, D.: It's a streaming world! reasoning upon rapidly changing information. IEEE Intelligent Systems **24**(6) (2009) 83–89
2. Gebser, M., Grote, T., Kaminski, R., Schaub, T.: Reactive answer set programming. In: Proceedings of LPNMR, Springer (2011) 54–66
3. Janhunen, T., Oikarinen, E., Tompits, H., Woltran, S.: Modularity aspects of disjunctive stable models. Journal of Artificial Intelligence Research **35** (2009) 813–857
4. Babb, J., Lee, J.: Action language $\mathcal{BC}$+: Preliminary report. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). (2015)
5. Giunchiglia, E., Lee, J., Lifschitz, V., McCain, N., Turner, H.: Nonmonotonic causal theories. Artificial Intelligence **153(1–2)** (2004) 49–104
6. Lee, J., Lifschitz, V., Yang, F.: Action language $\mathcal{BC}$: Preliminary report. In: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI). (2013)
7. Babb, J., Lee, J.: Module theorem for the general theory of stable models. TPLP **12**(4-5) (2012) 719–735
8. Ferraris, P.: Answer sets for propositional theories. In: Proceedings of LPNMR. (2005) 119–131
9. Bartholomew, M., Lee, J.: Stable models of multi-valued formulas: Partial vs. total functions. In: Proceedings of International Conference on Principles of Knowledge Representation and Reasoning (KR). (2014) 583–586
10. Ferraris, P., Lee, J., Lifschitz, V., Palla, R.: Symmetric splitting in the general theory of stable models. In: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), AAAI Press (2009) 797–803
11. Cerexhe, T., Gebser, M., Thielscher, M.: Online agent logic programming with oclingo. In: PRICAI 2014: Trends in Artificial Intelligence. Springer (2014) 945–957