Macros, Reactive Plans and Compact Representations

Christer Bäckström[1] and **Anders Jonsson**[2] and **Peter Jonsson**[1]

Abstract. The use and study of compact representations of objects is widespread in computer science. AI planning can be viewed as the problem of finding a path in a graph that is implicitly described by a compact representation in a planning language. However, compact representations of the path itself (the plan) have not received much attention in the literature. Although both macro plans and reactive plans can be considered as such compact representations, little emphasis has been placed on this aspect in earlier work. There are also compact plan representations that are defined by their access properties, for instance, that they have efficient random access or efficient sequential access. We formally compare two such concepts with macro plans and reactive plans, viewed as compact representations, and provide a complete map of the relationships between them.

1 INTRODUCTION

The concept of representations of objects that are much smaller than the objects themselves is widespread in computer science; we use it on a daily basis when we compress or uncompress files. Such representations are often referred to as compact, compressed or succinct. In some cases it is sufficient to compute a compact representation, for instance, when archiving a file. In other cases the representation must support various operations, like searching for or replacing data, without first unpacking it. Performing an operation on a compact representation of an object is typically harder than performing the same operation on the original object. This is not always the case, though; there are algorithms which are efficient because they use compact representations. Two such examples are the use of succinct representations of visibility regions to efficiently answer visibility questions for polygons [7], and succinct representations of solution sets for efficient solving of certain CSP problems [9]. Also AI planning can be tractable in certain cases by exploiting compact representations of the solutions [16, 19]. That is, using compact representations is sometimes beneficial even if not motivated by saving space.

An archetypical example of compact representations is compression of strings, with results varying from optimizing the size of string representations [12, 25] to efficient implementations of operations on compressed strings [4, 18]. Compact representations of more structured objects than strings is also well studied in the literature. For instance, compact representations of graphs have been studied both in the general case [14] as well as in connection with graph search motivated by AI applications [3]. Compact representations have been studied in numerous other AI applications such as model checking [23] and for analysing formalisms for knowledge representation [11]. AI planning is the application in focus of this paper.

[1] IDA, Linköping University, SE-581 83 Linköping, Sweden. Email: christer.backstrom@liu.se peter.jonsson@liu.se

[2] DTIC, Universitat Pompeu Fabra, 08018 Barcelona, Spain. Email: anders.jonsson@upf.edu

Planning has a natural connection to compact representations, but one which is often overlooked. A planning instance is an implicit representation of a graph that is exponentially larger than its representation and where the solutions, i.e. the plans, are paths in this graph. That is, the problem instances themselves are compact representations by definition. Yet, very little attention has been paid to understanding and analysing compact representations also of the solutions, which are usually explicitly represented. This is a surprising asymmmetry, especially since many plan representations that can be viewed as compact have been proposed in the literature; examples include macros, reactive plans, and contingent plans. Even though the inventors themselves may not always have realized that these representations are compact or invented them for that purpose, some of them do serve this purpose perfectly well.

The main goal of this paper is to analyse and compare a number of such representations taken from the literature. Although we argue that saving space is not the only, or even the most important, aspect of compact representations, it is not irrelevant even today. An obvious case is when the hardware is severly restricted, for one reason or another, as in the case of many automotive or autonomous-agent applications, cf. reactive plans in spaceships [27]. It is less obvious that compact representations are highly relevant even when we consider computing that is not hampered by severe hardware limitations. However, also this is sometimes the case, at least if we do not draw a rigid borderline between planning and search in general; Korf [21] has very recently considered disk-based search algorithms and argued that representational compactness is highly relevant for search. Another case is the use of large databases of plans or subplans, or even just some piece of information for each plan, as in pattern databases. These are used for heuristic search in planning and can become so big that they need to be compressed [13, 26], yet they do not even store the actual plans but only a heuristic value for each one! Clearly, compact databases are useful and important also for case-based reasoning and many other memory-intensive applications.

Apart from the obvious purpose of saving space there are other, often more interesting, reasons for considering compact representations. One reason is that compact representations can emphasize what different plans or subplans have in common and how they differ from each other. This might be exploited for more efficient reasoning about plans, for instance, by abstracting actions with similar function into equivalence classes [2]. Basically, compactness means structure, which is a well-known information-theoretic fact. If an object can be represented compactly, then it has some redundancy and structure that might be possible to exploit for simpler and more efficient reasoning. Furthermore, planning has a long tradition of inventing new planning languages and plan representations. While there are comparisons of languages in the literature, either from a knowledge-representation viewpoint or from a computational viewpoint, very little such work exists on comparing plan representations. We advocate

studying plan representations from the perspective of compactness as one of several ways for doing such comparative work, motivated by the relationship between compactness and structure.

To narrow in to more concrete examples of compact plan representations, macros is an obvious starting point. They have been widely used in planning for a long time, but seldom for this purpose. An exception is Giménez and Jonsson [16, 19] who study classes of planning problems that may have solutions of exponential length, but where a macro representation of a solution can always be generated in polynomial time. Macro plans are intimately related to the concept of compressed grammars, thus having close ties with the previously mentioned work on compact string representations. There are also compact representations that are characterised by their access properties, either efficient random access (CRAR [2] and TA [22]) or efficient sequential access (CSAR [2] and SA [22]). These are all different ways of representing one long plan compactly.

Another case is representing a large set of plans compactly, rather than one single long plan. This occurs in plan recognition, where we may have to consider an exponential number of candidate plans that share the same initial prefix [15]. Another example is reactive plans (also known as universal plans or reactive systems). Although seldom described in such terms, a reactive plan is actually a representation of a (usually large) set of plans: there is one plan for each state from which the goal can be reached. In both these examples, there will typically be a lot of redundant information, for instance, in the form of plan segments that are shared between several different plans. Representing such plan sets in a more compact way can be beneficial and, in large real-world examples, absolutely necessary. This is also how it is often done in practice; for instance, a reactive plan is often compactly described as an algorithm or a state machine.

A third case is representing a single plan that is big, but not necessarily long. A typical example of this is contingent plans [6] (or other branching plans) where each branch may be of moderate length but the number of branches may grow exponentially. Also here, we can often expect that different branches share sub-branches which can be exploited as described for sets of plans.

These three cases are not distinct and isolated from each other. For instance, representing a single plan compactly or representing a set of plans compactly can sometimes be viewed as two different sides of the same coin. This idea has been used for obtaining non-trivial complexity results, cf. Bäckström and Jonsson [2] or Liberatore [22]. Another example is when a contingent plan, which is a branching plan, is represented as one long conformant plan, which does not branch, as described by Bonet [5]. Hence, studying compact representations of a single plan is not as limited and restrictive as it may seem. It should also be noted that although this paper is restricted to STRIPS planning, this is not necessary for compact representations to be important and interesting. For instance, solutions to POMDPs may be thought of as a probabilistic variant of reactive plans and compactness of representations is important also in this case [8]. Furthermore, an HTN planning instance can alternatively be viewed as a grammar that expands to the set of plans for itself.

This paper focuses on four compact representations of plans: macro plans, CRARs, CSARs, and reactive plans. The main result is a full investigation of how these four types of representations relate to each other, which essentially results in a number of subclass and separation results. The results can be summarized and illustrated as in Figure 1. Loosely speaking, $X \sqsubset_p Y$ (or $X \sqsubset_p^\forall Y$) means that representation Y is a strictly more expressive compact representation than X (formal definitions follow later). For instance, we see that macro plans (MACR) are a strict subclass of CRAR. This means

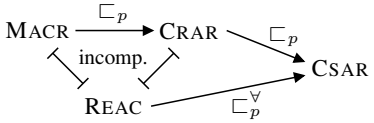

Figure 1. Summary of results.

that there is no plan that has a compact macro representations but does not have a CRAR. On the other hand, since the relationship is strict there must be plans that have compact representations with polynomial-time random access, yet do not have any compact macro representation. Thus, there are classes of plans that have more structure than macros can express but that still have CRARs. This says that it is not pointless to look for compact representations that are more expressive than macro plans, but that are still clearly structured and admit polynomial-time random access. One attempt in that direction is the concept of automata plans [1], which are highly structured and strictly more expressive than macro plans, although their relationship to CRARs remains to be precisely determined. We also see, for instance, that reactive plans are incomparable with both macro plans and CRARs. Among other things, this implies that they cannot be represented with macros or admit polynomial-time random access and still be compact; they are fundamentally different from both macro plans and CRARs, since the latter can represent things that reactive plans cannot. In short, the results tell us that all four types of representation have sufficiently different properties that none of them can be considered redundant. With these results we thus start to formally stake out a space of plan representations with varying properties.

These results are based on the one hand on a type of subclass relationship, and on the other hand on separation results. For the separation results it has been necessary to use a number of quite different techniques. Some of these have previously been used by Liberatore [22], but his techniques are not sufficient in all cases, so we also had to invent substantially different ones. For instance, we sometimes prove separation by demonstrating that plan validation has different complexity depending on the representation. This reveals a somewhat surprising connection between plan validation and compactness, which seems fruitful to investigate further. Among other things, this opens up for entirely new methods to classify planning problems into different complexity classes based on the properties of compact solutions for them. Some further discussion about this can be found in Section 6. Finally, the different techniques we have used provide a toolbox of tested methods that will most likely be useful also for investigations and comparisons of other plan representations.

The rest of the paper is organized as follows. Section 2 introduces some general notation and terminology, while Section 3 contains formal definitions of the four compact representations and the subsumption relations used to express their relationships. Section 4 analyses the complexity of plan validation for macro plans and CRARs, which is needed for the main results but is also interesting in its own right. Section 5 contains the main theorem along with some further results required. The paper ends with a discussion in Section 6.

2 PRELIMINARIES

The number of objects in a set or sequence X is denoted $|X|$ and the size of the representation of an object X is denoted $||X||$. Sequence concatenation is denoted ";" and X^* is the set of all sequences, in-

cluding the empty one, over a set X. Function composition is denoted $f \circ g$, where $(f \circ g)(x) = f(g(x))$. DTM and NTM denote deterministic and nondeterministic Turing machines, respectively. We use $\overline{x}$ to denote the negation of a propositional atom x, which extends to literals such that $\overline{\overline{x}} = x$ and to sets of literals such that $\overline{Y} = \{\overline{\ell} \mid \ell \in Y\}$. If X is a set of atoms, then $L(X) = \{x, \overline{x} \mid x \in X\}$, i.e. the literals over X. Let X be a universe of atoms. Then a set $Y \subseteq L(X)$ is *consistent* if either $x \notin L(X)$ or $\overline{x} \notin L(X)$ for all $x \in X$ and Y is *total* if exactly one of x and $\overline{x}$ is in Y for every $x \in X$. The closed-world assumption is thus made explicit, which is a purely techincal matter. The operator $\ltimes$ is defined as $X \ltimes Y = (X - \overline{Y}) \cup Y$ for all consistent sets X and Y of literals.

We use *propositional* STRIPS *with negative goals (PSN)*.

Definition 1. A *PSN frame* is a tuple $\boldsymbol{f} = \langle V, A \rangle$ where V is a set of propositional atoms and A is a set of actions. The *state space* is $S(\boldsymbol{f}) = \{s \in 2^{L(V)} \mid s \text{ total}\}$ and its members are called *states*. Each action a in A has a precondition $\mathrm{pre}(a) \subseteq L(V)$ and an effect $\mathrm{eff}(a) \subseteq L(V)$, which are both consistent. The notation $a : X \Rightarrow Y$ defines an action a with $\mathrm{pre}(a) = X$ and $\mathrm{eff}(a) = Y$. For all $s, t \in S(\boldsymbol{f})$ and actions $a \in A$: 1) a is *valid* in s if $\mathrm{pre}(a) \subseteq s$ and 2) action a is *from s to t* if a is valid in s and $t = s \ltimes \mathrm{eff}(a)$. A sequence $\omega = \langle a_1, \ldots, a_\ell \rangle \in A^*$ is a *plan* from s_0 to s_ℓ if either 1) $\omega = \langle\rangle$ and $s_0 = s_\ell$ or 2) there are $s_1, \ldots, s_{\ell-1} \in S(\boldsymbol{f})$ such that a_i is from s_{i-1} to s_i for all i, where $1 \leq i \leq \ell$. A *PSN instance* is a tuple $\boldsymbol{p} = \langle V, A, I, G \rangle$ such that $\boldsymbol{f} = \langle V, A \rangle$ is a PSN frame, $I \in S(\boldsymbol{f})$ and $G \subseteq L(V)$ is consistent. A *plan for $\boldsymbol{p}$* is a plan from I to some $s \in S(\boldsymbol{f})$ such that $G \subseteq s$,

3 COMPACT REPRESENTATIONS

This section starts with a brief recapitulation of macro plans and their access properties as well as of the CRAR and CSAR concepts. Macro plans are also cast as a compact representation in an analogous way and a subsumption relation is defined for comparing the expressive power of such representations. This is followed by a recapitulation of reactive plans and a generalisation of the subsumption relation that makes it possible to compare reactive plans, which are actually sets of plans, with representations of single plans.

A macro is a symbol with a definition that is a sequence of elements, where each element is either an action or a macro. A macro is expanded by replacing it with its defining sequence. A macro plan is a system of macros with acyclic expansion and a designated root macro, such that the root macro can be recursively expanded into a single well defined action sequence. This is essentially identical to a compressed grammar, which implies that macro plans have efficient random access properties.

Proposition 2. *(Bäckström and Jonsson [2], Proposition 31) There is a polynomial p such that for every PSN frame $\langle V, A \rangle$ and every macro plan μ for a sequence $\omega \in A^*$, μ can be used to random access any action in ω in $p(||\mu||)$ time.*

We now recapitulate the definition of the CRAR and CSAR concepts [2] and define a similar concept for macro plans. A DTM M runs with *delay* f, for some function f, if for all inputs x it generates each succesive output symbol in $f(||M|| + ||x||)$ time.

Definition 3. Let f be an arbitrary function, let $\boldsymbol{f} = \langle V, A \rangle$ be an arbitrary PSN frame and let $\omega = \langle a_1, \ldots, a_\ell \rangle \in A^*$. Then a *representation* ρ of ω is a DTM. Furthermore:

1) ρ is *f-compact* if $||\rho|| \leq f(||\boldsymbol{f}||)$ and it runs in $f(||\boldsymbol{f}||)$ space.

2) ρ is an *f-compact sequential-access representation* (f-CSAR) of ω if it is f-compact and it generates ω sequentially with delay $f(||\rho||)$. When finished, ρ outputs $\perp$ and then terminates.

3) ρ is an *f-compact random-access representation* (f-CRAR) of ω if it is f-compact and for an arbitrary index i (where $1 \leq i \leq |\omega|$) as input, it outputs action a_i of ω in time $f(||\rho||)$. Furthermore, ρ returns $\perp$ for all input not in the range $[1, |\omega|]$.

4) ρ is an *f-macro representation* (p-MACR) for ω if it is an f-compact macro plan for ω.

That ρ runs in $f(||\boldsymbol{f}||)$ space means the total of input, working and output tapes (which implies that an f-CRAR ρ cannot represent a plan longer than $2^{f(||\rho||)}$). We consider the output tape as cleared between actions so the output is a single action, not the sequence ω. We write only MACR, CRAR etc. when referring to the whole family of representations of a particular type. We proceed to define a subsumption relation that can be used to describe the relationship between different plan representations.

Definition 4. Let X and Y be representations of PSN action sequences. Then: $X \sqsubseteq_p Y$ if there is a polynomial-time function g such that for all PSN frames $\boldsymbol{f} = \langle V, A \rangle$ and all $\omega \in A^*$, if ρ is an X representation of ω then $g(\rho)$ is a Y representation of ω.

Note that the size of $g(\rho)$ is polynomially bounded in the size of ρ since g runs in polynomial time. We further write $X \sqsubset_p Y$ when $X \sqsubseteq_p Y$ holds but not $Y \sqsubseteq_p X$. While it may seem overly strong to require that we can transform representations in polynomial time and not just look at the size, this enables proving more results. Not all proofs make use of the time requirement, though.

A planning algorithm computes a whole plan from an intial state to a goal state while a *reactive plan* takes a state as input and outputs a single action to execute in that state, thus generating a plan incrementally, action by action. This definition follows Jonsson et al. [20].

Definition 5. A PSN *goal frame* is a tuple $\boldsymbol{g} = \langle V, A, G \rangle$ such that $\langle V, A \rangle$ is a PSN frame and $G \subseteq L(V)$ is consistent. Let $\boldsymbol{g} = \langle V, A, G \rangle$ be a PSN goal frame. Define $S_{ext} = S(\langle V, A \rangle) \cup \{\perp, \top\}$ and $A_{ext} = A \cup \{a_\perp, a_\top\}$. The virtual actions $a_\perp$ and $a_\top$ are defined such that for all states $s \in S_{ext}$: 1) $a_\perp$ is valid in s if $s \neq \top$ and $s \ltimes \mathrm{eff}(a_\perp) = \perp$. 2) $a_\top$ is valid in s if $s \neq \perp$ and $s \ltimes \mathrm{eff}(a_\top) = \top$. A *reactive plan* for $\boldsymbol{g}$ is a function $\rho : S_{ext} \to A_{ext}$. Define the corresponding function $\sigma_\rho : S_{ext} \to S_{ext}$ such that $\sigma_\rho(s) = s \ltimes \mathrm{eff}(\rho(s))$ for all $s \in S_{ext}$. The reactive plan ρ is:

Sound if for all $s \in S_{ext}$, 1) $\rho(s)$ is valid in s and 2) $\rho(s) = a_\top$ iff either $G \subseteq s$ or $s = \top$.

Acceptance-complete if for all $s \in S$ such that $\langle V, A, s, G \rangle$ has a plan there is an integer k such that $\sigma_\rho^k(s) = \top$.

Rejection-complete if for all $s \in S$ such that $\langle V, A, s, G \rangle$ has no plan there is an integer k such that $\sigma_\rho^k(s) = \perp$.

For all $s \in S_{ext}$, define $\rho^0(s) = \langle\rangle$ and $\rho^k(s) = (\,\rho(s); \rho^{k-1}(\sigma_\rho(s))\,)$ for all $k \geq 1$. Further define $\pi_\rho = \{\langle s, \rho^k(s) \rangle \mid k > 0,\ \sigma_\rho^k(s) \in S(\boldsymbol{g}) \text{ and } \sigma_\rho^{k+1}(s) \in \{\perp, \top\}\}$.

The function σ_ρ aggregates ρ with applying its result to the current state. A reactive plan encodes exactly one action sequence for each state and the set π_ρ is the set of all such sequences, keyed with their initial states. We specialize this general definition into a concept of compact representation, REAC, similar to the previous ones, which essentially captures $P_{T,S}AR$ universal plans [20] in the case where f is a polynomial. We also define a generalisation of the $\sqsubseteq_p$ relation since a REAC is a set of plans.

Definition 6. Let f be an arbitrary function and let $\boldsymbol{g} = \langle V, A, G\rangle$ be an arbitrary PSN goal frame. Then an f-REAC ρ is an f-compact reactive plan for $\boldsymbol{g}$ that is sound, acceptance-complete, rejection-complete and runs in $f(||\rho||)$ time.

Definition 7. Let X be an action-sequence representation. Then:

1) REAC $\sqsubseteq_p^{\forall} X$ if there is a polynomial-time function g such that for all PSN goal frames $\boldsymbol{g} = \langle V, A, G\rangle$, if ρ is a REAC for $\boldsymbol{g}$ then $g(\rho, s)$ is an X representation of ω for every $\langle s, \omega\rangle \in \pi_\rho$.

2) $X \; {}^{\forall}\!\sqsubseteq_p$ REAC if there is a polynomial p such that for all functions f and for all PSN goal frames $\boldsymbol{g} = \langle V, A, G\rangle$, if every $\langle V, A, s, G\rangle$ with a plan has an f-X for some plan then $\boldsymbol{g}$ has a $(p \circ f)$-REAC.

3) X and REAC are *incomparable* if neither REAC $\sqsubseteq_p^{\forall} X$ nor $X \; {}^{\forall}\!\sqsubseteq_p$ REAC holds.

The definition of $\sqsubseteq_p^{\forall}$ is analogous to $\sqsubseteq_p$ but requires that we can generate one X representation for every action sequence the REAC can generate. However, ${}^{\forall}\!\sqsubseteq_p$ has no obvious analogous definition since there can be many different plans from a state to the goal but a REAC can represent at most one plan for each state.

4 PLAN REPRESENTATION VALIDATION

This section analyses the complexity of validating a plan that is given as either a MACR or a CRAR. These results are required later to prove separation between the two concepts, although they are also interesting as stand-alone results. Especially important is the result that macro plans allow for efficient plan validation, in addition to efficient random access. Plan validation for arbitrary type R of plan representation is defined as follows.

Plan Validation for R
INSTANCE: A PSN instance $\boldsymbol{p} = \langle V, A, I, G\rangle$ and an R-representation ρ of a sequence $\omega \in A^*$.
QUESTION: Is ω a plan for $\boldsymbol{p}$?

The complexity of validation is measured in $||\boldsymbol{p}|| + ||\rho||$.

A macro is commonly treated as a compound action described by its cumulative precondition and effect, which is sufficient information to describe the macro [17], as follows.

Definition 8. Let $\langle V, A\rangle$ be a PSN frame. Then *consistency* and *cumulative precondition and effect* for action sequences is defined as follows: 1) $\langle\rangle$ is consistent and $\mathrm{pre}(\langle\rangle) = \mathrm{eff}(\langle\rangle) = \varnothing$. 2) Let $\omega \in A^*$ and $a \in A$. Then: a) $(\omega; a)$ is consistent if both ω and $\mathrm{eff}(\omega) \cup \mathrm{pre}(a)$ are consistent, b) $\mathrm{pre}(\omega; a) = \mathrm{pre}(\omega) \cup (\mathrm{pre}(a) - \mathrm{eff}(\omega))$ and c) $\mathrm{eff}(\omega; a) = \mathrm{eff}(\omega) \ltimes \mathrm{eff}(a)$.

Proposition 9. *Let $\boldsymbol{f} = \langle V, A\rangle$ be a PSN frame, $\omega \in A^*$ and $s, t \in S(\boldsymbol{f})$. Then ω is a plan from s to t if and only if all the following holds: 1) ω is consistent, 2)* $\mathrm{pre}(\omega) \subseteq s$ *and 3)* $t = s \ltimes \mathrm{eff}(\omega)$.

Hence, the root of a macro plan describes the whole plan, which enables us to prove the following result.

Theorem 10. *Plan Validation for* MACR *is in* **P**.

Proof sketch. Let $\boldsymbol{p} = \langle V, A, I, G\rangle$ be a PSN instance and let μ be a MACR with root macro r for some sequence $\omega \in A^*$. According to Proposition 9 it is sufficient to check that the expansion of r is consistent, that $\mathrm{pre}(r) \subseteq I$ and that $I \ltimes \mathrm{eff}(r) \subseteq G$ to decide whether μ represents a plan for $\boldsymbol{p}$ or not. It is straightforward from Definition 8 that there is a polynomial-time algorithm for computing consistency and cumulative conditions for all macros in a macro plan. Validating I and G against r is obviously no harder than this preprocessing. □

We will next prove that validation is harder for CRAR than for MACR but first need some further machinery. We define a family of generic PSN instances, then prove that this family corresponds to the class $\boldsymbol{\Pi}_2^p$ in the polynomial hierarchy and that these instances always have a plan with a polynomial CRAR.

Construction 11. Let $F = \forall x_1, \ldots, x_m \exists y_1, \ldots, y_n \,.\, \phi$ be a $\forall\exists$-3SAT formula where $\phi = (c_1 \wedge \ldots \wedge c_h)$ and $c_i = \ell_i^1 \vee \ell_i^2 \vee \ell_i^3$ is a 3-literal clause for all i such that $1 \leq i \leq h$. Construct the PSN instance $\boldsymbol{p}_F = \langle V, A, I, G\rangle$ such that $V = \{mix, miy, mvc, mvl, okx, oky, x_1, \ldots, x_m, y_1, \ldots, y_n, vc_0, \ldots, vc_h\}$, $I = \{\overline{mix}, \overline{miy}, mvc, \overline{mvl}, \overline{x_1}, \ldots, \overline{x_m}, \overline{y_1}, \ldots, \overline{y_n}, \overline{vc_0}, \ldots, \overline{vc_h}, \overline{okx}, \overline{oky}\}$, $G = \{miy, x_1, \ldots, x_m\}$ and A has the following actions:

$abv\colon \{mvc, \overline{vc_0}\} \Rightarrow \{vc_0, oky\}$

$avc_j\colon \{mvc, \overline{mvl}, vc_{j-1}, \overline{vc_j}\} \Rightarrow \{mvl, vc_j\}$
$avt_j^1\colon \{mvl, vc_j, \ell_j^1\} \Rightarrow \{\overline{mvl}\}$
$avt_j^2\colon \{mvl, vc_j, \overline{\ell_j^1}, \ell_j^2\} \Rightarrow \{\overline{mvl}\}$
$avt_j^3\colon \{mvl, vc_j, \overline{\ell_j^1}, \overline{\ell_j^2}, \ell_j^3\} \Rightarrow \{\overline{mvl}\}$
$avf_j\colon \{mvl, vc_j, \overline{\ell_j^1}, \overline{\ell_j^2}, \overline{\ell_j^3}\} \Rightarrow \{\overline{mvl}, \overline{oky}\}$

$aet\colon \{mvc, \overline{mvl}, vc_h, oky\} \Rightarrow \{miy, \overline{mvc}, \overline{vc_0}, \ldots, \overline{vc_h}, okx\}$
$aef\colon \{mvc, \overline{mvl}, vc_h, \overline{oky}\} \Rightarrow \{miy, \overline{mvc}, \overline{vc_0}, \ldots, \overline{vc_h}\}$

$aiy_i\colon \{miy, \overline{y_i}, y_{i-1}, \ldots, y_1\} \Rightarrow \{\overline{miy}, mvc, y_i, \overline{y_{i-1}}, \ldots, \overline{y_1}\}$
$ary\colon \{miy, y_n, \ldots, y_1, okx\} \Rightarrow \{mix, \overline{miy}, \overline{y_n}, \ldots, \overline{y_1}, \overline{okx}\}$

$aix_i\colon \{mix, \overline{x_i}, x_{i-1}, \ldots, x_1\} \Rightarrow \{\overline{mix}, mvc, x_i, \overline{x_{i-1}}, \ldots, \overline{x_1}\}$

Let $X(i)$ be the binary encoding of i using the x variables and $Y(i)$ analogous. Define the action sequence ω_F hierarchically such that

$$\begin{array}{lcl} \omega_F & = & E_0, aix, E_1, aix, \ldots, aix, E_{2^m-1} \\ E_i & = & V_i^0, aiy, V_i^1, aiy, \ldots, aiy, V_i^{2^n-1}, ary \\ V_i^j & = & abv, avc_1, avx_1, avc_2, avx_2, \ldots, avc_h, avx_h, aex \end{array}$$

where $1 \leq i \leq 2^m - 1$ and $1 \leq j \leq 2^n - 1$. For each i, the aix action following E_i is aix_k where k is the largest number k' such that $i \mod 2^{k'-1} = 0$. For each i and j, the aiy action following V_i^j is aiy_k where k is the largest number k' such that $j \mod 2^{k'-1} = 0$. For each V_i^j, $aex = aet$ if $c_1, \ldots, c_h$ are all satisfied in $X(i) \cup Y(j)$ and otherwise $aex = aef$. For each V_i^j and each k, action avx_k is

$$avx_k = \begin{cases} avt_k^1 & \text{if } \ell_k^1 \in X(i) \cup Y(j) \\ avt_k^2 & \text{if } \{\overline{\ell_k^1}, \ell_k^2\} \subseteq X(i) \cup Y(j) \\ avt_k^3 & \text{if } \{\overline{\ell_k^1}, \overline{\ell_k^2}, \ell_k^3\} \subseteq X(i) \cup Y(j) \\ avf_k & \text{otherwise} \end{cases}$$

Lemma 12. *There is a polynomial p such that for every $\forall\exists$-3SAT formula F, with $\boldsymbol{p}_F$ and ω_F as defined in Construction 11: 1) ω_F is a plan for $\boldsymbol{p}_F$ if and only if F is satsifiable and 2) ω_F has a p-CRAR ρ_F that can be constructed in polynomial time from $\boldsymbol{p}$.*

The proof is omitted, but the crucial part of it is the algorithm in Figure 2, which is a polynomial CRAR satisfying the claims for ρ_F, where $L_V = 2h + 2$ is the length of a V_i^j block and $L_E = 2^n(L_V + 1)$ is the length of an E_i block.

Note that ω_F always exists for $\boldsymbol{p}_F$ but is not always a plan. Furthermore, despite its clear hierarchical structure and regularity, ω_F is not an obvious candidate for a macro plan. The reason is that each E_i block contains an exponential number of V_i^j blocks that might differ so much that it is unlikely that we can always represent them with a polynomial number of macros. We now have the necessary prerequisites to prove the complexity of CRAR validation.

function $\rho_F(t)$
  $i := \lfloor \frac{t-1}{L_E+1} \rfloor$, $o_E := t - i(L_E+1)$
  $j := \lfloor \frac{o_E-1}{L_V+1} \rfloor$, $o_V := o_E - j(L_V+1)$
  if $t < 1$ **or** $t > 2^m(L_E+1) - 1$ **then** $a = \bot$
  elsif $o_E = L_E + 1$ **then** *(action is type aix)*
    $k := \max\{k' \mid i \mod 2^{k'-1} = 0\}$, $a := aix_k$
  else *(action is in E_i)*
    if $o_V = L_V + 1$ **then** *(action is type aiy or ary)*
      if $j = L_V + 1$ **then** $a := ary$
      else $k := \max\{k' \mid j \mod 2^{k'-1} = 0\}$, $a := aiy_k$
    else *(action is in V_i^j)*
      $k := (o_V - 1)/2$
      if $o_V = 1$ **then** $a := abv$
      elsif $o_V = L_V + 1$ **then**
        if $c_1, \ldots, c_h$ are all satisfied in $X(i) \cup Y(j)$ **then**
          $a := aet$
        else $a := aef$
      elsif o_V is odd **then** $a := avc_k$
      elsif $\ell_k^1 \in X(i) \cup Y(j)$ **then** $a := avt_k^1$
      elsif $\{\overline{\ell_k^1}, \ell_k^2\} \subseteq X(i) \cup Y(j)$ **then** $a := avt_k^2$
      elsif $\{\overline{\ell_k^1}, \overline{\ell_k^2}, \ell_k^3\} \subseteq X(i) \cup Y(j)$ **then** $a := avt_k^3$
      else $a := avf_k$
  return a

Figure 2. Algorithm for the CRAR ρ_F.

Theorem 13. *Plan Validation for p-CRAR is 1) in $\mathbf{\Pi}_2^p$ for all polynomials p and 2) $\mathbf{\Pi}_2^p$-hard for all polynomials $p \in \Omega(n^c)$ for some constant c.*

Proof sketch. (1 Membership): Let p be an arbitrary polynomial. Before proving the main result, we consider the following problem X.

INSTANCE: A PSN instance $\boldsymbol{p} = \langle V, A, I, G\rangle$, a p-CRAR for some sequence $\omega = \langle a_1, \ldots, a_n\rangle \in A^*$, a literal $\ell \in L(V)$ and two integers i, j such that $1 < i < j < 2^{p(||\boldsymbol{p}||)}$.

QUESTION: Is there an integer k s.t. $i < k < j$ and $\ell \in \mathrm{eff}(a_k)$?

X is in **NP** since it is sufficient to guess a k between i and j and verify that $\ell \in \mathrm{eff}(a_k)$. Hence, the complementary problem C of deciding if a literal does not change between a_i and a_j is in co**NP**.

To prove the main result, first consider its complementary problem, deciding if ρ does not encode a plan for $\boldsymbol{p}$. If ω is not a plan for $\boldsymbol{p}$, then there must be two integers i and j and a literal ℓ such that 1) $i < j$, 2) $\ell \in \mathrm{pre}(a_j)$, 3) $\overline{\ell} \in \mathrm{eff}(a_i)$ and 4) $\ell \notin \mathrm{eff}(a_k)$ for all k such that $i < k < j$. Verify that ω is not a plan by using an NTM with an oracle for C as follows. First guess i, j and ℓ. Check that conditions 1–3 are satisfied and then use the oracle to verify condition 4, which is an instance of C. Since C is in co**NP** this whole procedure is a problem in $\mathbf{NP}^{\mathrm{co}\mathbf{NP}}$. Hence, the complementary problem of verifying that ω is a plan is in $\mathrm{co}\mathbf{NP}^{\mathrm{co}\mathbf{NP}} = \mathrm{co}\mathbf{NP}^{\mathbf{NP}} = \mathbf{\Pi}_2^p$. The initial state and goal are handled similarily.

(2 Hardness): Let p be a polynomial satisfying Lemma 12. Let F be an arbitrary $\forall\exists$-3SAT formula and ω_F as in Definition 11. Lemma 12 then says there is a p-CRAR ρ_F for ω_F. The tuple $\langle \boldsymbol{p}_F, \rho_F\rangle$ is an instance of Plan Representation Validation for p-CRAR and it follows from Lemma 12 that ω_F is a plan for $\boldsymbol{p}$ if and only if F is satisfiable. Furthermore, since we can compute both $\boldsymbol{p}_F$ and ρ_F for arbitrary F in polynomial time, there is a polynomial reduction from $\forall\exists$-3SAT to Plan Representation Validation for p-CRAR. Hence Plan Representation Validation for p-CRAR is $\mathbf{\Pi}_2^p$-hard since $\forall\exists$-3SAT is $\mathbf{\Pi}_2^p$-hard. □

If we do not know if a string is a CRAR over an action set, then we must check that first. We leave it without proof that this problem is in co**NP** and, thus, does not add to the complexity of plan validation.

5 RELATIONSHIP RESULTS

We now finally prove some further necessary results and then head for the main theorem, which formally states the relationship results earlier claimed.

Lemma 14. *All of* MACR $^{\forall}\not\sqsubseteq_p$ REAC, CRAR $^{\forall}\not\sqsubseteq_p$ REAC *and* CSAR $^{\forall}\not\sqsubseteq_p$ REAC *hold unless PH collapses.*

Proof sketch. Jonsson et. al. [20, Lemma 12–13 and Theorem 14] construct a generic PSN goal frame $\boldsymbol{g}_n = \langle V_n, A_n, G_n\rangle$ for every positive integer n, and prove that there is no polynomial p such that for all $n > 0$, $\boldsymbol{g}_n$ has a reactive plan ρ_n that is acceptance-complete, p-compact and runs in in $p(||\boldsymbol{g}_n||)$ time. However, they also show that for all $n > 0$ and all $s \in S(\boldsymbol{g}_n)$, if $\langle V_n, A_n, s, G_n\rangle$ has a plan then this has at most $8n^3 + 2n$ actions. □

Lemma 15. REAC $\not\sqsubseteq_p^{\forall}$ CRAR *unless PH collapses.*

Proof sketch. Bylander [10, Theorem 3.1] demonstrated a polynomial reduction from **PSPACE** to PSN planning by constructing a PSN instance $\boldsymbol{p}$ for an arbitrary polynomial-space bounded DTM M and input x such that $\boldsymbol{p}$ has a plan if and only if $M(x)$ accepts. Furthermore, if $M(x)$ accepts in n steps, then the plan has $3n+1$ steps, and at most one action is applicable in any state. Hence, finding the right action in a state, if there is one, is polynomial-time. Since M is bounded there is a predictable limit k for the maximum number of machine steps if $M(x)$ accepts, and thus also a limit $3k+1$ for the plan length. It follows that there is a polynomial p such that for every $M(x)$, the corresponding $\boldsymbol{p}$ has a p-REAC.

Suppose REAC $\sqsubseteq_p^{\forall}$ CRAR. Let $M(x)$ be a polynomial-space bounded DTM with input, let $\boldsymbol{p} = \langle V, A, I, G\rangle$ be the corresponding PSN encoding of $M(x)$ and let ρ be a corresponding p-REAC as described above. Suppose $M(x)$ accepts. Then $\boldsymbol{p}$ has a plan ω of length $3k+1$ or shorter. By assumption, we can construct a CRAR for ω from ρ in time $q(||\rho||)$ for some fixed polynomial q, so ω has a $(q \circ p)$-CRAR. We can, thus, verify that $M(x)$ accepts by guessing a string w of size at most $(q \circ p)(||\boldsymbol{p}||)$ and verify that it is a plan for $\boldsymbol{p}$, which is in $\mathbf{\Pi}_2^p$ according to Theorem 13. Hence, finding a p-CRAR for some plan for $\boldsymbol{p}$ is in $\mathbf{NP}^{\mathbf{\Pi}_2^p} = \mathbf{\Sigma}_3^p$. However, this means that deciding if $M(x)$ accepts is in $\mathbf{\Sigma}_3^p$ so **PSPACE** $\subseteq \mathbf{\Sigma}_3^p$, which is impossible unless the polynomial hierarchy collapses. It follows that REAC $\not\sqsubseteq_p^{\forall}$ CRAR. □

Theorem 16. *If the polynomial hierarchy does not collapse, then: 1)* MACR $\sqsubset_p$ CRAR $\sqsubset_p$ CSAR*; 2)* REAC $\sqsubset_p^{\forall}$ CSAR*; 3)* CRAR *and* REAC *are incomparable; 4)* MACR *and* REAC *are incomparable.*

Proof of Theorem 16. 1) CRAR $\sqsubseteq_p$ CSAR and CSAR $\not\sqsubseteq_p$ CRAR are immediate from Bäckström and Jonsson [2, Theorems 29 and 30]; MACR $\sqsubseteq_p$ CRAR follows from Proposition 2 and Definition 3; while CRAR $\not\sqsubseteq_p$ MACR follows from Theorems 10 and 13. 2) REAC $\sqsubseteq_p^{\forall}$ CSAR is trivial and CSAR $^{\forall}\not\sqsubseteq_p$ REAC follows from Lemma 14. 3) Follows from Lemmas 14 and 15. 4) MACR $^{\forall}\not\sqsubseteq_p$ REAC follows from Lemma 14. For the opposite direction, suppose REAC $\sqsubseteq_p^{\forall}$ MACR. Then REAC $\sqsubseteq_p^{\forall}$ CRAR since MACR $\sqsubseteq_p$ CRAR. However, this contradicts Lemma 15 so REAC $\not\sqsubseteq_p^{\forall}$ MACR holds unless PH collapses. □

6 DISCUSSION

While it is possible to imagine other concepts than our subsumption concept for relating representations, we have tested the strength of it by achieving the results in this paper. It is furthermore a natural concept that has many similarities with Liberatore's concepts [22]. It is also worth noting that in some cases, like Lemma 14, we prove a much stronger form of separation than Definition 4 requires. Instead of proving that there is no polynomial function g from X to Y we prove that there does not even exist a polynomially bounded Y representation for every X representation. Liberatores results [22] do no immediately fit into our analysis. For instance, his separation results are weaker than ours since he uses a more powerful circuit-based action representation. Furthermore, although his TA representation is essentially identical to the CRAR concept, his SA representation resembles the CSAR concept but is more like a reactive plan that represents only a single plan.

The CRAR and CSAR concepts might, perhaps, seem very theoretical compared to concepts like macro plans and reactive plans, which have been frequently used in practice. There is no such clear distinction, however. A macro plan has a clear and simple structure. This makes it easy to use and understand, but at the expense of limited expressive power. Reactive systems, on the other hand, are often described as algorithms in one form or another, without much structure imposed on them in the general case. This is no different from describing a CRAR with an algorithm, as in this paper. It is, however, obvious that structured representations have many advantages and that more expressive ones than macro plans can be useful. A recent attempt in that direction is the concept of automata plans [1]. One should, thus, primarily consider CRAR and CSAR as yardsticks for the classification of other, more structured, representations.

Although the purpose of this paper is to compare the expressive power of some different types of compact representations, the results also hint at another possible use of this type of results. Plan representations and their properties open up for an entirely new way to define classes of planning problems. For instance, the class of planning instances that have polynomial-size macro plans as solutions is in **NP**, since plan validation is in **P** for macro plans. On the other hand, the class of planning instances that have polynomial-size CRARs as solutions cannot be in **P** unless the polynomial hierarchy collapses. That is, the latter class is more expressive. This is related to the compilation-based classification by Nebel [24], but he primarily studies how various features in a planning languages affect the size of instances, rather than the size of plans.

In this paper we only consider compact plan representations that are exact representations of some explicit plan. However, it is also common to use plans which are compact because they are not exact representations. An example is abstraction, where the planning process uses an abstraction of the search space to guide the planning, either by explicitly constructing a plan in this abstract space or by somehow implicitly computing a heuristic value for it. This abstract plan is typically shorter and easier to compute, but sacrifices certain correctness criteria. We believe that further study of the relationship between such inexact compact plans and exact compact ones may cast new light on abstraction.

It should finally be noted that although we draw a separating line between representations that are polynomial and those that are not, this is just a first coarse classification. In the future it might be interesting to make similar studies of relationships and separations between representations of different polynomial degrees.

ACKNOWLEDGEMENTS

A. Jonsson is partially supported by grants TIN2009-10232, MICINN, Spain, and EC-7PM-SpaceBook.

REFERENCES

[1] C. Bäckström, A. Jonsson, and P. Jonsson, 'From macro plans to automata plans', in *20th European Conf. Artif. Intell. (EACI'12), Montpellier, France*, (2012).
[2] C. Bäckström and P. Jonsson, 'Algorithms and limits for compact plan representations', *J.Artif. Intell. Res.*, **44**, 141–177, (2012).
[3] J. Balcázar, 'The complexity of searching implicit graphs', *Artif. Intell.*, **86**(1), 171–188, (1996).
[4] P. Bille, G. Landau, R. Raman, K. Sadakane, S. Satti, and O. Weimann, 'Random access to grammar-compressed strings', in *22'nd ACM-SIAM Symp. Discrete Algorithms (SODA'11), San Fransisco, CA, USA*, pp. 373–389, (2011).
[5] B. Bonet, 'Conformant plans and beyond: Principles and complexity', *Artif. Intell.*, **174**(3-4), 245–269, (2010).
[6] B. Bonet and H. Geffner, 'Planning with incomplete information as heuristic search in belief space', in *5th Int'l Conf. Artif. Intell. Planning Systems (AIPS'00), Breckenridge, CO, USA*, pp. 52–61, (2000).
[7] P. Bose, A. Lubiw, and I. Munro, 'Efficient visibility queries in simple polygons', *Comput. Geom.*, **23**(3), 313–335, (2002).
[8] C. Boutilier and D. Poole, 'Computing optimal policies for partially observable decision processes using compact representations', in *13th Nat'l Conf. Artif. Intell. (AAAI'96), Portland, OR, USA*, volume 2, pp. 1168–1175, (1996).
[9] A. Bulatov and V. Dalmau, 'A simple algorithm for Mal'tsev constraints', *SIAM J. Comput.*, **36**(1), 16–27, (2006).
[10] T. Bylander, 'The computational complexity of propositional STRIPS planning', *Artif. Intell.*, **69**(1-2), 165–204, (1994).
[11] M. Cadoli, F. Donini, P. Liberatore, and M. Schaerf, 'Space efficiency of propositional knowledge representation formalisms', *J.Artif. Intell. Res.*, **13**, 1–31, (2000).
[12] M. Charikar, E. Lehman, D. Liu, R. Panigrahy, M. Prabhakaran, A. Sahai, and A. Shelat, 'The smallest grammar problem', *IEEE Trans. Information Theory*, **51**(7), 2554–2576, (2005).
[13] A. Felner, R. Korf, R. Meshulam, and R. Holte, 'Compressed pattern databases', *J. Artif. Intell. Res.*, **30**, 213–247, (2007).
[14] H. Galperin and A. Wigderson, 'Succinct representations of graphs', *Information and Control*, **56**(3), 183–198, (1983).
[15] C. Geib, 'Assessing the complexity of plan recognition', in *19th Nat'l Conf. Artif. Intell. (AAAI'04), San José, CA, USA*, pp. 507–512, (2004).
[16] O. Giménez and A. Jonsson, 'The complexity of planning problems with simple causal graphs', *J.Artif. Intell. Res.*, **31**, 319–351, (2008).
[17] P. Haslum and P. Jonsson, 'Planning with reduced operator sets', in *5th Int'l Conf. Artif. Intell. Planning Systems (AIPS'00), Breckenridge, CO, USA*, pp. 150–158, (2000).
[18] J. Jansson, K. Sadakane, and W-K. Sung, 'Compressed random access memory', *ArXiv*, **abs/1011.1708v2**, (2012).
[19] A. Jonsson, 'The role of macros in tractable planning', *J. Artif. Intell. Res.*, **36**, 471–511, (2009).
[20] P. Jonsson, P. Haslum, and C. Bäckström, 'Towards efficient universal planning: A randomized approach', *Artif. Intell.*, **117**(1), 1–29, (2000).
[21] Richard E. Korf, 'Linear-time disk-based implicit graph search', *J. ACM*, **55**(6), (2008).
[22] P. Liberatore, 'Complexity issues in finding succinct solutions of PSPACE-complete problems', *ArXiv*, **abs/cs/0503043**, (2005).
[23] P. Liberatore and M. Schaerf, 'On the size of data structures used in symbolic model checking', *ArXiv*, **abs/1012.3018**, (2010).
[24] B. Nebel, 'On the compilability and expressive power of propositional planning formalisms', *J. Artif. Intell. Res.*, **12**, 271–315, (2000).
[25] W. Rytter, 'Application of Lempel-Ziv factorization to the approximation of grammar-based compression', *Theor. Comput. Sci.*, **302**(1-3), 211–222, (2003).
[26] E. Schreiber and R. Korf, 'Using partitions and superstrings for lossless compression of pattern databases', in *Proc.25th AAAI Conf. Artif. Intell., (AAAI'11), San Francisco, CA, USA*, (2011).
[27] B. Williams and P. Pandurang Nayak, 'A reactive planner for a model-based executive', in *15th Int'l Joint Conf. Artif. Intell. (IJCAI'97), Nagoya, Japan*, pp. 1178–1185, (1997).