A Polynomial All Outcome Determinization for Probabilistic Planning

Thomas Keller and **Patrick Eyerich**
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Georges-Köhler-Allee 52
79110 Freiburg, Germany
{tkeller,eyerich}@informatik.uni-freiburg.de

Abstract

Most predominant approaches in probabilistic planning utilize techniques from the more thoroughly investigated field of classical planning by determinizing the problem at hand. In this paper, we present a method to map probabilistic operators to an equivalent set of probabilistic operators in a novel normal form, requiring polynomial time and space. From this, we directly derive a determinization which can be used for, e. g., replanning strategies incorporating a classical planning system. Unlike previously described all outcome determinizations, the number of deterministic operators is not exponentially but polynomially bounded in the number of parallel probabilistic effects, enabling the use of more sophisticated determinization-based techniques in the future.

Introduction

Most probabilistic planning systems that have successfully participated in the International Probabilistic Planning Competition, e.g. FF-Replan (Yoon, Fern, and Givan 2007), FPG (Buffet and Aberdeen 2007) or RFF-(BG/PG) (Teichteil-Königsbuch, Infantes, and Kuter 2008), invoke a procedure called *determinization* of the probabilistic planning task.

Two classes of determinization strategies have been described: *Single outcome* determinizations choose one possible outcome for each probabilistic operator, accepting that solvable tasks might become unsolvable in the determinization, while *all outcome* determinizations preserve solvability by generating all potential outcomes. The only all outcome determinization used in practice generates one operator for each potential outcome, possibly leading to exponentially many operators in the determinization (Rintanen 2003). While Rintanen briefly mentions a solution to this problem, it has never been described in detail, a shortfall made up for with this paper by introducing the *forked normal form* (FNF) and a polynomial determinization based on FNF where operators are *split* into several deterministic ones that are applied *sequentially* to simulate one outcome.

After presenting our formal framework, we show how to split operators while preserving task equivalence, and how to generate a set of operators in FNF with size polynomial in the number of probabilistic effects. We continue by discussing the derived determinization, and finally point out the benefit with a short experiment.

Probabilistic Planning

Definition 1. A (fully-observable) probabilistic *planning task* T is a 4-tuple $\langle V, s_0, S_\star, O\rangle$, where V is a set of *finite-domain state variables* v with *domain* D_v; s_0, the *initial state*, is a valuation over V; $S_\star$, the *goal*, is a *condition* over V, a logical formula over atoms of the form $v{=}d$ ($v \in V, d \in D_v$); and O is a set of *operators*, 3-tuples $o = \langle \text{pre}(o), \text{eff}(o), c(o)\rangle$ with *precondition* $\text{pre}(o)$, a condition over V; *effect* $\text{eff}(o)$; and *cost* $c(o) \in \mathbb{N}_0$. Effects of operators $o \in O$ are recursively defined by:

- $\top$ is the *empty* effect.
- $v{\leftarrow}x$ with $v \in V, x \in D_v$ is an *atomic* effect.
- $e_1 \wedge e_2$ is an *conjunctive* effect if e_1 and e_2 are effects.
- $c \rhd e$ is a *conditional* effect if c is a condition and e is an effect.
- $p_1 e_1 | \ldots | p_n e_n$ is a *probabilistic* effect if $e_1, \ldots, e_n$ are effects, $0 < p_1, \ldots, p_n \leq 1$, and $\sum_{i=1}^{n} p_i = 1$.

An effect e without probabilistic effects is *deterministic*. An effect e is in *unary conditionality normal form* (UCNF) if e' is atomic for all $c \rhd e'$ in e; it is in *unary non-determinism normal form* (1ND) if e is of the form $p_1 e_1 | \ldots | p_n e_n$ with deterministic $e_1, \ldots, e_n$ (Rintanen 2003), the normal form the most common all outcome determinization is based on. We prohibit effects that assign multiple values to the same v as semantics might become ambiguous. To measure plan quality we use costs, but our results also hold for planning problems based on rewards (Condon 1992). In the following, we refer to valuations s over V as the *states* of T.

The *successor set* $\text{suc}(s, o)$ of applying an operator o in a state s is a set of state/probability pairs defined as in Rintanen's definition of operator application (2003).

Definition 2. A probabilistic *transition system* $\mathcal{T}$ is a 5-tuple $\langle S, O, \Delta, s_0, G\rangle$ where S is a set of *states*; O is a set of operators; for each $o \in O$ there is a $\delta_o \in \Delta$ that is a partial function mapping states to probability distributions over S; $s_0 \in S$ is the initial state; and $G \subseteq S$ is the set of goal states. States with more than one applicable operator are called *decision states* $\text{dec}(\mathcal{T})$, and $\text{suc}_{\text{dec}}(s) = (\text{suc}(s) \cap \text{dec}(\mathcal{T})) \cup \{\text{suc}_{\text{dec}}(s') | s' \in \text{suc}(s) \setminus \text{dec}(\mathcal{T})\} \subseteq \text{dec}(\mathcal{T})$ is the recursively defined set of *decision successors*.

A planning task $T = \langle V, s_0, S_\star, O\rangle$ induces a transition system $\mathcal{T} = \langle S, O, \Delta, s_0, G\rangle$, where S is the set of states of

V; for each $o \in O$ there is a $\delta_o \in \Delta$ defined for all states s where $s \models \text{pre}(o)$ mapping s to a probability distribution induced by $\text{suc}(s, o)$; and $G = \{s \in S | s \models S_\star\}$. A *policy* π is a mapping from states to operators. We refer to the set of all policies in a transition system $\mathcal{T}$ as $\Pi(\mathcal{T})$.

Since we compare planning tasks where the application of an operator in one task is matched to the application of a sequence of operators in the other (i.e. there is no one-to-one correspondence between transitions), we need our definition of equivalence to be more general than usual. We achieve this by exploiting the fact that a policy π naturally induces a probability distribution over costs $\alpha_\pi(s)$ for each state s:

Definition 3. Two transition systems $\mathcal{T}_1$ and $\mathcal{T}_2$ are *equivalent* ($\mathcal{T}_1 \equiv \mathcal{T}_2$) if there are two bijective mappings $\sigma : \text{dec}(\mathcal{T}_1) \to \text{dec}(\mathcal{T}_2)$ and $\theta : \Pi(\mathcal{T}_1) \to \Pi(\mathcal{T}_2)$ s.t. for all $\pi \in \Pi(\mathcal{T}_1)$ and $s \in \text{dec}(T_1)$ we have $\alpha_\pi(s) = \alpha_{\theta(\pi)}(\sigma(s))$.

Operator Splitting

A determinization based on 1ND leads to exponentially many operators if conjunctions of probabilistic effects are present. The main idea of our determinization is to split these, creating one operator per conjunctive element. In this section, we show how to split deterministic operators and enforce their sequential application, and how to prevent non-equivalent transformation due to conditional effects.

Operator splitting leads to *intermediate* states in the transformed task where only parts of the original operator's effects have been applied. To control correct operator application we introduce a variable v_{aux} and extend all preconditions of operators such that $v_{\text{aux}}{=}0$ is requested. In intermediate states we require different, unique values for v_{aux} and thereby enforce sequential application in an equivalence-preserving way for effects not containing conditional effects. Note that intermediate states are no decision states due to the uniqueness of the values for v_{aux}, a property that also holds in the rest of this paper.

Example 1. An operator $o = \langle v_0{=}2, v_1{\leftarrow}4 \wedge v_2{\leftarrow}0, c(o)\rangle$ can be split into $o'_1 = \langle v_0{=}2 \wedge v_{\text{aux}}{=}0, v_1{\leftarrow}4 \wedge v_{\text{aux}}{\leftarrow}1, c(o)\rangle$ and $o'_2 = \langle v_{\text{aux}}{=}1, v_2{\leftarrow}0 \wedge v_{\text{aux}}{\leftarrow}0, 0\rangle$.

This transformation can lead to a non-equivalent task if a variable v occurs both in an effect condition c and an atomic effect, and if that operator is split s.t. the assignment of v is applied before c is checked. To avoid this, we show how to transform a planning task $T = \langle V, s_0, S_\star, O\rangle$ into an equivalent planning task $T^\triangleright = \langle V \cup V^\triangleright \cup \{v_{\text{aux}}\}, \sigma(s_0), S_\star \cup \{v_{\text{aux}}{=}0\}, O^\triangleright \cup O^\star\rangle$, where no variable occurs both in effect conditions and atomic effects.

Theorem 1. *All planning tasks can be normalized in polynomial time and space to equivalent planning tasks where no variable occurs both in effect conditions and atomic effects.*

Proof: Let $\# : O \to \{1, \ldots, |O|\} \subset \mathbb{N}$ be bijective, $V^\triangleright$ be a set of variables with one variable $v_\triangleright$ for each $v \in V$, and the operators $o_\triangleright = \langle \text{pre}(o_\triangleright), \text{eff}(o_\triangleright), c(o)\rangle$ and $o_\star = \langle \text{pre}(o_\star), \text{eff}(o_\star), 0\rangle$ for each $o \in O$ be s.t.

$$\text{pre}(o_\triangleright) = \text{pre}(o) \wedge v_{\text{aux}}{=}0$$
$$\text{eff}(o_\triangleright) = \bigwedge_{v \in V} v_\triangleright{\leftarrow}\text{val}(v) \wedge v_{\text{aux}}{\leftarrow}\#(o)$$
$$\text{pre}(o_\star) = v_{\text{aux}}{=}\#(o)$$
$$\text{eff}(o_\star) = \text{eff}(o)[V/V^\triangleright] \wedge v_{\text{aux}}{\leftarrow}0$$

where $\text{val}(v) \in D_v$ is the value of v and $\text{eff}(o)[V/V^\triangleright]$ is equal to $\text{eff}(o)$ except that all occurrences of all $v \in V$ in effect conditions are replaced by their corresponding $v_\triangleright \in V^\triangleright$. As polynomial time and space transformation from T to $T^\triangleright$ is obvious we focus on the proof of equivalence with

$$\sigma(s) = s \cup \{(v, 0) | v \in V^\triangleright \cup \{v_{\text{aux}}\}\}, \text{ and}$$
$$\theta(\pi)(\sigma(s)) = o_\triangleright \Leftrightarrow \pi(s) = o$$

in the following. To show that σ is bijective, it is sufficient to show that $\sigma(s_1) \neq \sigma(s_2)$ unless $s_1 = s_2$ for $s_1, s_2 \in \text{dec}(\mathcal{T})$ and that $\text{dec}(\mathcal{T}^\triangleright) \subseteq \{\sigma(s) | s \in \text{dec}(\mathcal{T})\}$. The former follows directly from the definition of σ, while the latter is not as obvious: We prove it by showing that there is no $s_\triangleright \in \text{dec}(\mathcal{T}^\triangleright)$ for which there is no $s \in \text{dec}(\mathcal{T})$ with $\sigma(s) = s_\triangleright$. Following the definition of σ, we must thus show that $s_\triangleright(v) = 0$ for all $v \in V^\triangleright \cup \{v_{\text{aux}}\}$ and all $s_\triangleright \in \text{dec}(\mathcal{T}^\triangleright)$. Let $s_\triangleright \in S^\triangleright$ be a state with $s_\triangleright(v) \neq 0$ for some $v \in V^\triangleright \cup \{v_{\text{aux}}\}$. The definition of $\text{pre}(o_\triangleright)$ directly shows that $s_\triangleright \not\models \text{pre}(o_\triangleright)$ for all $o_\triangleright \in O^\triangleright$. Additionally, if $s_\triangleright \models \text{pre}(o_\star)$ it follows that $s_\triangleright \not\models \text{pre}(o'_\star)$ for $o_\star, o'_\star \in O_\star$ by definition of $\#(o)$ and $\text{pre}(o_\star)$. For this reason there is at most one operator o with $s_\triangleright \models o$, and thus $s_\triangleright \notin \text{dec}(\mathcal{T}^\triangleright)$.

The bijectivity of θ on $\text{dec}(T)$ and $\text{dec}(T^\triangleright)$ is given as $\pi(s_\triangleright) \notin O_\star$ for all $s_\triangleright \in \text{dec}(\mathcal{T}^\triangleright)$ because $s_\triangleright \not\models o_\star$ for all $o_\star \in O_\star$ for all policies π. To verify that $T \equiv T^\triangleright$, we first show that for each policy π in $\mathcal{T}$ it holds that $\alpha_\pi(s) = \alpha_{\theta(\pi)}(\sigma(s))$ for all states $s \in \text{dec}(\mathcal{T})$, which is given iff for all $s \in \text{dec}(\mathcal{T})$ and all sequences of operators leading from s to any $s' \in \text{suc}_{\text{dec}}(s)$, there is a sequence of operators from $\sigma(s)$ to $\sigma(s')$ that induces the same probability and cost (and vice versa). Let $o^1, \ldots, o^k \in O$ be the sequence of operators to reach an s' from s with probability p and cost c. Then the sequence $o^1_\triangleright, o^1_\star, \ldots, o^k_\triangleright, o^k_\star$ leads from $\sigma(s)$ to $\sigma(s')$ with probability p, as the $o_\triangleright$ are deterministic and the transition probabilities from the $o_\star$ and o are equal, and cost c, as $c(o) = c(o_\triangleright) + c(o_\star)$ by definition of the cost functions of $o_\triangleright$ and $o_\star$. As all sequences of operators in $T^\triangleright$ from any $s_\triangleright \in \text{dec}(\mathcal{T})$ to any $s'_\triangleright \in \text{suc}_{\text{dec}}(s_\triangleright)$ must be of the form $o^1_\triangleright, o^1_\star, \ldots, o^k_\triangleright, o^k_\star$ by definition of $\text{eff}(o_\triangleright)$, $\text{pre}(o_\star)$ and $\#(o)$, the opposite direction holds as well. ∎

Forked Normal Form

So far we restricted our analysis to deterministic operators, but the procedure to normalize operators with conditional effects is applicable to operators with probabilistic effects as well, resulting in a set of deterministic operators $O^\triangleright$ and a set of arbitrary operators $O^\star$. Without loss of generality, we also assume that $\text{eff}(o_\star)$ is in UCNF for all $o_\star \in O^\star$ (possible in polynomial time and space: Rintanen 2003), and that all instances of the left-hand side of the equivalence

$$p_1(p'_1 e'_1 | \ldots | p'_j e'_j) | \ldots | p_i e_i \equiv p_1 p'_1 e'_1 | \ldots | p_1 p'_j e'_j | \ldots | p_i e_i$$

are additionally replaced by the right-hand side (obviously possible in polynomial time and space), resulting in effects of the form

$$\bigwedge_i p_{i1} e_{i1} | \ldots | p_{in_i} e_{in_i} \wedge \bigwedge_j e_j,$$

where the e_{ik} are effects of that form themselves and the e_j are conditional effects $e_j = c \rhd e'$ with atomic effects e'.

Definition 4. An effect e is in *forked normal form* (FNF) if it is deterministic or of the form

$$(p_1 e_1 | \dots | p_i e_i | \dots | p_n e_n) \wedge \bigwedge_j e_j,$$

where all e_i and e_j are conditional effects $c \rhd e'$ with atomic effects e'. An operator o is in FNF if $\text{eff}(o)$ is in FNF.

While there are effects that cannot be transformed into a *single* effect in FNF, it is interesting nevertheless as there is a polynomial time algorithm to transform a planning task $T^{\rhd}$ (created from an arbitrary planning task T) into an equivalent planning task $T^{\Psi} = \langle V \cup V^{\rhd} \cup \{v_{\text{aux}}\}, \sigma(s_0), S_\star \cup \{v_{\text{aux}}{=}0\}, O^{\rhd} \cup O^{\Psi} \rangle$, where all operators are in FNF, and the number of operators is polynomially bounded in the number of probabilistic effects in $o_\star \in O^\star$.

All $o_\Psi \in O^\Psi$ that are created by Algorithm 1 from $o_\star \in O^\star$ have 3 important commonalities besides being in FNF:

- $\text{pre}(o_\Psi) = v_{\text{aux}}{=}\#(o_\Psi)$, where $\#(o_\Psi)$ is unique.
- $\text{eff}(o_\Psi) = \bigwedge_i e_i \wedge (p_1 v_{\text{aux}}{\leftarrow}x_1 | \dots | p_n v_{\text{aux}}{\leftarrow}x_n)$, where $x_1, \dots, x_n \in \mathbb{N}$ are pairwise distinct.
- $c(o) = 0$ (costs are covered by the procedure described earlier).

To create an operator o_Ψ we need to know its deterministic effects e_i, its index $\#(o_\Psi)$, the probabilities p_i, and the values x_i assigned to v_{aux}. While the first three pieces of information are known when an effect is met the first time (lines 2–4), the last is not available before all children have been created as their index corresponds to these values. Due to this, Algorithm 1 maintains a directed acyclic graph of FNF-nodes that correspond to the operators in FNF, as depicted in Figure 1. Once all children have been generated recursively (lines 6–8) they are appended to all leaves (lines 9–10) and o_Ψ is added (line 11).

Theorem 2. *Each planning task can be normalized in polynomial space and time to an equivalent planning task where all operators are in FNF.*

Proof sketch: Our procedure creates one operator for each probabilistic outcome, needing polynomial space, and is obviously computable in polynomial time. Equivalence of $T^{\rhd}$ and T^{Ψ} can be shown analogously to the proof of Theorem 1 by using the same mappings σ and θ and the same properties of decision states. Due to limited space, we sketch the algorithm's behavior in an example instead. ∎

Example 2. Consider the following $\text{eff}(o_\star)$ for $o_\star \in O^\star$:

$$\big(0.5(s{\leftarrow}0 \wedge (0.3\, t{\leftarrow}1 \mid 0.7\, u{\leftarrow}3) \wedge (0.6\, v{\leftarrow}0 \mid 0.4\, w{\leftarrow}4)) \mid 0.5(x{\leftarrow}1)\big) \wedge (0.5\, y{\leftarrow}2 \mid 0.5\, z{\leftarrow}1) \wedge q{\leftarrow}2$$

Let 1 be the first generated index. The effects in FNF, created from the directed acyclic graph in Figure 1, include:

$$\begin{aligned}
\text{eff}(o^1_\Psi) &= q{\leftarrow}2 \wedge (0.5 v_{\text{aux}}{\leftarrow}2 | 0.5 v_{\text{aux}}{\leftarrow}7) \\
\text{eff}(o^2_\Psi) &= s{\leftarrow}0 \wedge (0.3 v_{\text{aux}}{\leftarrow}3 | 0.7 v_{\text{aux}}{\leftarrow}4) \\
\text{eff}(o^3_\Psi) &= t{\leftarrow}1 \wedge (0.6 v_{\text{aux}}{\leftarrow}5 | 0.4 v_{\text{aux}}{\leftarrow}6) \\
\text{eff}(o^4_\Psi) &= u{\leftarrow}3 \wedge (0.6 v_{\text{aux}}{\leftarrow}5 | 0.4 v_{\text{aux}}{\leftarrow}6) \\
&\dots \\
\text{eff}(o^9_\Psi) &= z{\leftarrow}1 \wedge v_{\text{aux}}{\leftarrow}0
\end{aligned}$$

Algorithm 1: Algorithm creating a set of operators in FNF.

```
1  def buildFNF(effect)
2      if effect is conditional then
3          return FNFNode({effect},nextIndex())
4      result = FNFNode(effect.detEffs(),nextIndex())
5      for eff in effect.probEffs() do
6          children = {}
7          for out in eff.outcomes() do
8              children.add(⟨buildFNF(out), eff.prob(out)⟩)
9          for leaf in result.leaves() do
10             leaf.setChildren(children)
11             createOperator(leaf,children)
12     return result
```

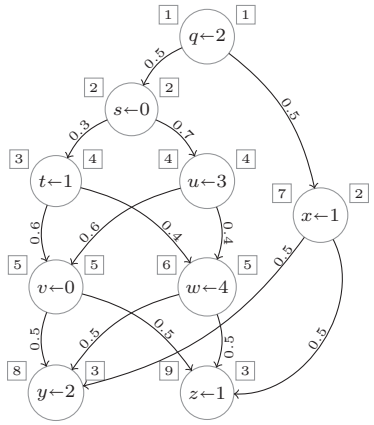

Figure 1: Directed acyclic graph created by Algorithm 1 for $\text{eff}(o_\star)$ in Example 2, each FNF-node corresponding to an operator in FNF and containing its deterministic effects. Numbers in the upper left corner denote the values of v_{aux} in preconditions, and labels give transition probabilities. Numbers in the upper right corner denote the values of v_{aux} in preconditions in the determinization.

Determinization

A deterministic planning task is a planning task in which all operators are deterministic. Our definition of a determinization, which transforms a probabilistic planning task to a deterministic one, captures the same semantics as that of a *compilation* by Little and Thiébaux (2007).

Definition 5. A sequence $t = \langle o_0, \dots, o_n \rangle$ of operators $o_i \in O$ is a *trajectory* in $\mathcal{T} = \langle S, O, \Delta, s_0, G \rangle$, if o_0 is applicable in s_0 and there are states $s_i \in S$ s.t. $\delta_{o_{i-1}}(s_{i-1})(s_i) > 0$ for $1 \le i \le n+1$ and $s_{n+1} \in G$ and o_i is applicable in s_i. The *cost* of t is $\text{cost}(t) = \sum_{i=0}^{n} c(o_i)$.

Definition 6. A planning task $T^d = \langle V^d, s^d_0, S^d_\star, O^d \rangle$ is a *determinization* of a planning task $T = \langle V, s_0, S_\star, O \rangle$ if it is deterministic and for each trajectory t^d in $\mathcal{T}^d$ there is a trajectory t in $\mathcal{T}$ s.t. $\text{cost}(t^d) = \text{cost}(t)$. If there also is a trajectory t^d in $\mathcal{T}^d$ for each trajectory t in $\mathcal{T}$ s.t. $\text{cost}(t) = \text{cost}(t^d)$, T^d is an *all outcome* determinization of T.

Given a planning task T^Ψ as created by Algorithm 1, we can directly generate an all outcome determinization T^d of T^Ψ. Due to limited space we only give a brief description

of the determinization process. Operators $o_{\triangleright} \in O^{\triangleright}$ are already deterministic, and all operators $o_{\Psi} \in O^{\Psi}$ are either deterministic or it holds that $\text{pre}(o_{\Psi}) = v_{\text{aux}}{=}\#(o_{\Psi})$ and $\text{eff}(o_{\Psi}) = \bigwedge_i e_i \wedge (p_1 v_{\text{aux}}{\leftarrow}x_1 | \ldots | p_n v_{\text{aux}}{\leftarrow}x_n)$. Note that assignments of the form $v_{\text{aux}}{\leftarrow}0$ only occur in deterministic operators which remain unchanged.

Let E_p be the set that contains the probabilistic part e_p of $\text{eff}(o_{\Psi})$ for all $o_{\Psi} \in O^{\Psi}$, i.e., assignments of the form $(p_1 v_{\text{aux}}{\leftarrow}x_1 | \ldots | p_n v_{\text{aux}}{\leftarrow}x_n)$ (note that e_p is equal for $o_{\Psi} \in O^{\Psi}$ that were created from FNF-nodes that share their children). Furthermore, let $\#_{\text{det}} : E_p \to \{1, \ldots, |E_p|\}$ be bijective. To determinize we simple replace the probabilistic part e_p of each operator $o_{\Psi} \in O^{\Psi}$ with the deterministic effect $v_{\text{aux}}{\leftarrow}\#_{\text{det}}(e_p)$, thereby creating a deterministic operator.

Example 3. In Figure 1, the numbers on the upper right corner of the nodes indicate the new values of v_{aux} when applying this procedure to Example 2, creating amongst others:

$$\text{eff}(o_d^1) = q{\leftarrow}2 \wedge v_{\text{aux}}{\leftarrow}2$$
$$\text{eff}(o_d^2) = s{\leftarrow}0 \wedge v_{\text{aux}}{\leftarrow}4$$
$$\text{eff}(o_d^3) = t{\leftarrow}1 \wedge v_{\text{aux}}{\leftarrow}5$$
$$\text{eff}(o_d^4) = u{\leftarrow}3 \wedge v_{\text{aux}}{\leftarrow}5$$

The resulting planning task T^d is an all outcome determinization of T^{Ψ} and thus, with our former results, of T.

Experiments

While the standard benchmarks of the latest IPPCs do not make intensive use of parallel probabilistic effects, we believe this feature to be significant in probabilistically interesting planning problems. One example is the Canadian Traveler's Problem (CTP), a path planning problem in an undirected graph where each edge has a weight and a probability of being traversable (Papadimitriou and Yannakakis 1991). Whether an edge is blocked or not is revealed to the agent only if it is located in an adjacent node, so the agent has to reason about the expected cost of paths. Recently, several high-quality domain-dependent strategies for solving the CTP have been proposed (Eyerich, Keller, and Helmert 2010). With the help of parallel probabilistic effects we can encode the CTP very concisely in PPDDL, requiring only one schematic operator changing both the agent's location and determining presence or absence of roads.

Table 1 compares the determinizations based on Rintanen's 1ND normal form (1NDD) and on the forked normal form proposed in this paper (FNFD). For three instances of the CTP with an increasing number of nodes and operators we denote the number of generated deterministic operators, the blowup factor relative to the number of probabilistic operators (indicated by ↑ in the table), and the average number of conditional or atomic effects in the generated operators.

It can be seen that FNFD generally generates far fewer operators than 1NDD. Since 1NDD is exponential in the number of parallel probabilistic effects it highly depends on average and maximal branching factor, which explains why the blowup factor in the problem with 50 nodes (standard deviation of 4.8) is larger than in the problem with 100 nodes (standard deviation of 3.7). In contrast to that, FNFD depends only linearly on the average number of parallel probabilistic effects.

Nodes/Ops	# Roads	Det. Ops	⌀Effects
20/98	4.9 ± 3.7	1122 ↑11.4	3.7
		6008 ↑61.3	21.9
50/278	5.56 ± 4.8	3550 ↑12.8	3.7
		29744 ↑107.0	25.3
100/568	5.68 ± 3.7	7348 ↑12.9	3.7
		55160 ↑97.0	23.7

Table 1: Determinizations based on 1ND (white) and FNF (light gray) on three instances of the CTP with different number of nodes, operators and adjacent roads (with the standard deviation). We state the number of generated deterministic operators (the blowup factor relative to the number of original operators is indicated by ↑) and the average number of effects of the generated operators.

Furthermore, the deterministic operators generated by 1NDD contain a lot of redundancy which is reflected in their average number of effects. In contrast to that, the operators generated by FNFD have a constant average number of effects (3.7) for all problems in the CTP.

Conclusion

In this paper, we have presented FNF, a normal form for probabilistic planning, and a polynomial time and space procedure to map planning tasks to equivalent tasks where all operators are in FNF. Each generated operator can directly be transformed into a single deterministic operator, thereby gaining an all outcome determinization without the exponential blowup resulting from previous determinizations. The results of our experiment emphasize the advantages of our FNF-based determinization.

We will further use FNF to implement a UCT-based domain-independent planning system, providing us with a second advantage: The structure of the graph depicted in Figure 1 can be used in the UCT rollouts to quickly generate nodes that correspond to intermediate states and where probabilistic outcomes are sampled sequentially, thereby avoiding the generation of the possibly exponentially many successor nodes when applying an operator that is not in FNF.

References

Buffet, O., and Aberdeen, D. 2007. FF + FPG: Guiding a policy-gradient planner. In *ICAPS*, 42–48.

Condon, A. 1992. The Complexity of Stochastic Games. *Information and Computation* 96(2):203–224.

Eyerich, P.; Keller, T.; and Helmert, M. 2010. High-Quality Policies for the Canadian Traveler's Problem. In *AAAI*, 51–58.

Little, I., and Thiébaux, S. 2007. Probabilistic Planning vs. Replanning. In *ICAPS Workshop on IPC*.

Papadimitriou, C. H., and Yannakakis, M. 1991. Shortest Paths Without a Map. *TCS* 84(1):127–150.

Rintanen, J. 2003. Expressive Equivalence of Formalisms for Planning with Sensing. In *ICAPS*, 185–194.

Teichteil-Königsbuch, F.; Infantes, G.; and Kuter, U. 2008. RFF: A Robust, FF-Based MDP Planning Algorithm for Generating Policies with Low Probability of Failure. IPPC Planner Abstract.

Yoon, S. W.; Fern, A.; and Givan, R. 2007. FF-Replan: A Baseline for Probabilistic Planning. In *ICAPS*, 352–359.