# Dealing with Continuous Resources in AI Planning

**Susanne Biundo** and **Roland Holzer** and **Bernd Schattenberg**
University of Ulm
Department of Artificial Intelligence
D-89069 Ulm, Germany
{biundo,holzer,schatten}@informatik.uni-ulm.de

**Abstract.** This paper presents an approach towards probabilistic planning with continuous resources. It adopts stochastic concepts for continuous probabilities and integrates them into a STRIPS-based planning framework. The approach enables the construction of plans that are guaranteed to meet certain probability thresholds w.r.t. the consumption of critical resources. Furthermore, the consumption probabilities of multiple resources can be accumulated and thus an overall probability for a successful execution of an aggregate plan can be computed. We extend our approach to HTN-based planning and show how heuristics can be derived that lead to plans with a minimized average value/variance of their overall resource consumption.

## 1 Introduction and Motivation

Autonomous agents that act in alien environments are exposed to a variety of unpredictable conditions and developments. These may lead to non-deterministic effects of the actions taken and may in particular cause some uncertainty w.r.t. resource consumption. The consumption of resources can be mission-critical, however. Autonomous spacecrafts, for example, have only a limited amount of propellant available and must carefully plan consuming activities in order to guarantee or maximize the mission's success. At present, the literature provides a variety of conformant and conditional planners, which are able to deal with discrete non-determinism (see e.g. (Bertoli *et al.* 2001; Goldman and Boddy 1994; Majercik and Littman 1998; Peot and Smith 1992; Weld *et al.* 1998)). If continuous resources, like energy, fuel, or propellant are involved, these approaches are often less suitable, however. The reason is that they require a (simplifying) discretized representation, which is often too imprecise for an informative prediction about resource consumption in actual real-world applications.

In this paper, we introduce an approach to handle uncertain consumptions of continuous resources in planning. We adopt the concepts for continuous probabilities and their computations from stochastics and integrate them into a standard STRIPS-based planning framework. The resulting probabilistic planning approach allows for an adequate representation of continuous resources and their consumption under uncertainty. The probabilities of resource consumptions of single actions and action sequences can be efficiently computed during planning. This enables the construction of plans that are guaranteed to meet certain probability thresholds w.r.t. the consumption of critical resources. Furthermore, the consumption probabilities of multiple resources can be accumulated. This allows for the computation of an overall probability for a successful execution of an aggregate plan. Not only does this enable a qualified decision if various alternative solutions are at hand, it even suggests a useful pruning of the search space. As a consequence, we extend our approach to HTN-based planning and show how heuristics can be generated semi-automatically that lead to the generation of plans with a minimized average value and/or variance of a particular resource consumption.

The rest of the paper is organized as follows. Section 2 introduces the basic notions of our probabilistic planning approach. In Section 3 it is shown how the computation of resource consumptions can be integrated into a simple planning framework. We extend our approach to HTN planning in Section 4 and show how the probabilities for resource consumptions of abstract tasks are accumulated across various decomposition methods and along the task hierarchy, respectively. Finally, we review related work in Section 5 and conclude with some remarks in Section 6.

## 2 Basic Definitions

Our planning formalism relies on the usual STRIPS representations of states and operators. A *state* is a finite set of ground atoms. An *operator* $o$ = (prec($o$), add($o$), del($o$)) consists of three such sets: the *preconditions* and the *positive* and *negative effects*, respectively. An operator $o$ is *applicable* in a state $s$ iff prec($o$) $\subseteq s$. The result of applying operator $o$ = (prec($o$), add($o$), del($o$)) in state $s$ is a state $result$($s$, $o$) = ($s \cup$ add($o$)) $\setminus$ del($o$).

A *plan* $p = \langle o_0 ... o_n \rangle$ is a sequence of operators such that for every state $s_i$, in which $o_i$ is applicable, we have that $o_{i+1}$ is applicable in $result(s_i, o_i)$, where $s_{i+1} = result(s_i, o_i)$ for $0 \leq i \leq n$. A plan $p$ is then *applicable* in $s_0$, and the resulting state

$result(result(...result(s_0, o_0), o_1)..., o_n)$ of $p$ is denoted by $result(s_0, p)$.

A *planning problem* is a triple $(O, I, G)$, where $O$ is a set of operators and $I$ and $G$ are sets of ground atoms, the *initial state* and the *goal*, respectively. A plan $p$ is a solution of such a planning problem iff $p$ is applicable in $I$ and $G \subseteq result(I, p)$ holds.

In stochastics, continuous events are modelled by continuous *random variables*. A *random variable* $\mathcal{X} : \Omega \rightarrow \mathbb{R}$ is a measurable function that maps the event space $\Omega$ onto the real numbers $\mathbb{R}$. The *distribution* of $\mathcal{X}$ is described by a *probability density* $D : \mathbb{R} \rightarrow \mathbb{R}$, denoted by $\mathcal{X}_D$ or $\mathcal{X} \sim D$.

The *mean value* $\mu = E(\mathcal{X}_D)$ of a random variable, with density $D$ is defined as $\int_{-\infty}^{+\infty} xD(x)dx$ and the *variance* $\mathrm{Var}(\mathcal{X}_D)$ of a random variable is given as $E((\mathcal{X}_D)^2) - E(\mathcal{X}_D)$. The *standard derivation* $\sigma_{\mathcal{X}}$ is defined by $\sqrt{\mathrm{Var}(\mathcal{X}_D)}$; i.e. the *variance* is also denoted by $\sigma^2$.

A *probability distribution* $F(\tau)$ over a probability density $D(x)$ is defined by the function $F(\tau) := \int_{-\infty}^{\tau} D(x)dx$. Given $F(\tau)$, probabilities for events can be computed as follows: $Pr[\mathcal{X}_D < a] = F(a)$, for example, is the probability of the "event", that the value of $\mathcal{X}_D$ is less then $a$. The event that the value $\mathcal{X}_D$ lies in the interval $(a..b)$ is described by $a < \mathcal{X}_D < b$, and the probability of this event is computed by $Pr[a < \mathcal{X}_D < b] = F(b) - F(a)$.

Random variables are specifications of resource consumptions. They represent rigid random functions and do not get bound to values like common logical variables. They are called *random* variables because the function value for a preimage is not fixed. For example, suppose that the life time of a MOS FET transistor is a random variable $\mathcal{X}$ with a mean value $\mu$ of 100 years and a variance $\sigma^2$ of 12 years. It is known that $\mathcal{X}$ is approximately normal-distributed*. The point in time when a specific transistor fuses cannot be predicted exactly but using the (known) distribution of $\mathcal{X}$ we can be sure by $99.8\%$ that the transistor works up to 90 years without a failure.

The first step to handle continuous uncertain resources in planning is to integrate random variables in the underlying representation formalism. To this end, we extend our STRIPS-based operator description by adding to each operator $o$ a set of random variables $rc(o)$. We also extend the definition of a state by adding a set of random variables to each state. Each such random variable represents the amount of a resource available in that state. $rc(o)$ is the *resource consumption* containing a random variable ${}_o\mathcal{X}^{r_i}$ for every resource $r_i \in \mathcal{R}$, where $\mathcal{R}$ is supposed to be a set of resource symbols. A random variable ${}_o\mathcal{X}^r$ specifies the consumption of resource $r$ of operator $o$. The density function $D$ of ${}_o\mathcal{X}^{r_i}$ defines the distribution, thereby reflecting the uncertainty of the resource consumption. If an operator $o$ does not consume a resource $r_i$ the respective random variable ${}_o\mathcal{X}^{r_i}$ has a *variance* and a *mean value* of zero.

Using random variables in planning has the advantage that we can compute the probability of an over-allocation of a resource prior to execution.

Often, interval arithmetic is used to represent an uncertain consumption. However, using interval arithmetic, one can only assert whether a plan will succeed (the sum of all upper bounds of the respective intervals is less than the available amount of the resource) or fail (the sum of the lower bounds exceeds the resource). We cannot make quantitative or qualitative predictions for the case between these extremes. Random variables, on the other hand, allow for exactly this kind of assertions.

## 3 Computing the Resource Consumption of a Plan

We assume that all random variables in a plan are stochastically independent. The probability density of a plan is calculated by the sum of all random variables in the plan. Suppose we are given a domain model with one resource, i.e. $\mathcal{R} = \{r\}$, and let the operator sequence $o_1, \ldots, o_n$ be a plan. ${}_{o_i}\mathcal{X}^r$ is the random variable which describes the consumption of resource $r$ by action $o_i$. The density of the sum of two random variables ${}_{o_i}\mathcal{X}^r_{D_1}$ and ${}_{o_j}\mathcal{X}^r_{D_2}$ is defined as the density $D_{{}_{o_i}\mathcal{X}^r_{D_1} + {}_{o_j}\mathcal{X}^r_{D_2}}$ which is calculated by *convolution*:

$$\begin{aligned} D_{{}_{o_i}\mathcal{X}^r_{D_1} + {}_{o_j}\mathcal{X}^r_{D_2}}(t) &:= \int_{-\infty}^{+\infty} D_1(\tau) D_2(t-\tau) d\tau \\ &= \int_{-\infty}^{+\infty} D_1(t-\tau) D_2(\tau) d\tau \end{aligned}$$

In the general case, the computation of a convolution is hard and time consuming. A restriction to normal-distributed random variables $\mathcal{X}_{\mathcal{N}_{(\mu,\sigma^2)}}$ relaxes this problem and speeds up the planning process. This is because if ${}_{o_i}\mathcal{X}^r$ are normal-distributed random variables with distributions $\mathcal{N}(\mu_{o_i,r}, \sigma^2_{o_i,r})$ (denoted as ${}_{o_i}\mathcal{X}^r \sim \mathcal{N}(\mu_{o_i,r}, \sigma^2_{o_i,r})$), the following holds:

$${}_{o_i}\mathcal{X}^r \sim \mathcal{N}(\mu_{o_i,r}, \sigma^2_{o_i,r}) : \sum_i {}_{o_i}\mathcal{X}^r \sim \mathcal{N}\left(\sum_i \mu_{o_i,r}, \sum_i \sigma^2_{o_i,r}\right)$$

The density of the sum of normal-distributed random variables can be computed efficiently, because only $2n$ real-valued additions are necessary to convolute $n$ densities $D_i$. The convolution is commutative and associative. This allows to compute the overall resource density during plan generation without constraining, e.g., the order in which operators are inserted.

The initial amount of a resource is modelled by a random variable with a positive mean value. This random variable

*The normal-distribution has the probability density $\varphi_{\mu,\sigma^2}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

can have a variance of zero if the amount of the resource is certain[†].

## An Example

Imagine a robot which is currently at position $A$ and has initially 10 units of energy. The goal of the robot's mission is to drive over the open field to a stone at location $B$ and get a sample for further analysis. A sample can be extracted by either using a drill or a milling cutter.

To get distributions for the resource consumption of the operators test data is used. The generated test samples are evaluated by, e.g., the *maximum likelihood estimation* (Daniels and Carrillo 1997) to construct approximative normal distributed densities.

**The Planning Problem** Suppose our domain model contains the three actions: `moveAtoB`, `milling`, and `drill`. In the following we often use as abbreviation mo, mi, and dr.

| | |
|---|---|
| `moveAtoB` | prec(mo) = {at(A)} |
| | add(mo)={at(B)}, del(mo)= {at(A)} |
| | res(mo)=$\{_{\text{mo}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(-4,1)}\}$ |
| `drill` | prec(dr) = {at(B)} |
| | add(dr)={hasSample}, del(dr)= {} |
| | res(dr)=$\{_{\text{dr}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(-4.5,2)}\}$ |
| `milling` | prec(mi) = {at(B)} |
| | add(mi)={hasSample}, del(mi)= {} |
| | res(mo)=$\{_{\text{mi}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(-4.65,0.1)}\}$ |

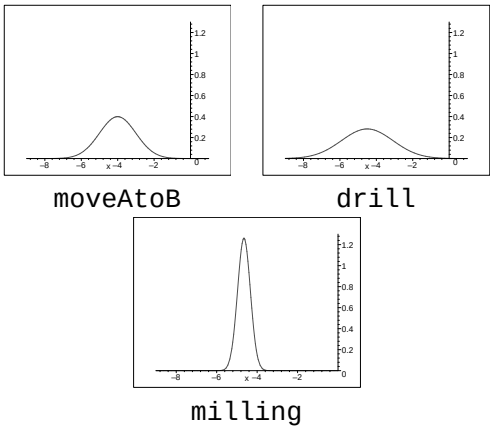


Figure 1: Energy densities of the random variables of the actions.

The initial state is given as `at(A)` with the initial resource value modelled by $_{\text{init}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(10,0)}$; $\sigma^2$ is zero in this case because this value is exactly known. The goal is $hasSample$. Figure 1 shows the energy densities of the actions described above.

[†]A normal-distribution with variance zero is a distribution, not a function. It is described by the constraints $\int_{-\infty}^{+\infty} f(x)dx = 1$ and $f(x) > 0 \quad \forall x$.

**The Planning Process** There are two possible plans to solve the problem. The first plan contains the actions `moveAtoB` and `drill`, the second `moveAtoB` and `milling`. All actions have an uncertain normal-distributed energy consumption.

The energy consumption of the first plan can be represented by the random variable $_{\text{(mo,dr)}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(-8.5,3)} = {}_{\text{mo}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(-4,1)} + {}_{\text{dr}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(-4.5,2)}$. To get the probability of a successful execution we have to compute the $Pr$ function of the event $X \geq 0$. The $Pr$ function maps events like $X < 0$ onto a probability.

$$\begin{aligned} &Pr[{}_{\text{init}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(10,0)} + {}_{\text{(mo,dr)}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(-8.5,3)} \geq 0] \\ &= Pr[{}_{\text{(init,mo,dr)}}\mathcal{Y}^{\text{energy}}_{\mathcal{N}(1.5,3)} \geq 0] \\ &= 1 - Pr[{}_{\text{(init,mo,dr)}}\mathcal{Y}^{\text{energy}}_{\mathcal{N}(1.5,3)} < 0] \\ &\approx 80.68\% \end{aligned}$$

This means, the plan will succeed with a probability of $80.68\%$. Suppose, the planner is given a threshold of 90% for the lowest acceptable probability, it will reject this plan and generate another one. This new plan has the two actions `moveAtoB` and `milling`. The probability of a successful execution is:

$$\begin{aligned} &Pr[{}_{\text{init}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(10,0)} + {}_{\text{mo}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(-4,1)} + {}_{\text{mi}}\mathcal{X}^{\text{energy}}_{\mathcal{N}(-4.65,0.1)} \geq 0] \\ &= 1 - Pr[{}_{\text{(init,mo,mi)}}\mathcal{Y}^{\text{energy}}_{\mathcal{N}(1.35,1.1)} \geq 0] \\ &\approx 90.1\% \end{aligned}$$

This probability is greater than the threshold and the plan can be accepted by the planner. Section 4 describes how a planner can be guided to search for plans with minimal $\mu$ and $\sigma^2$.

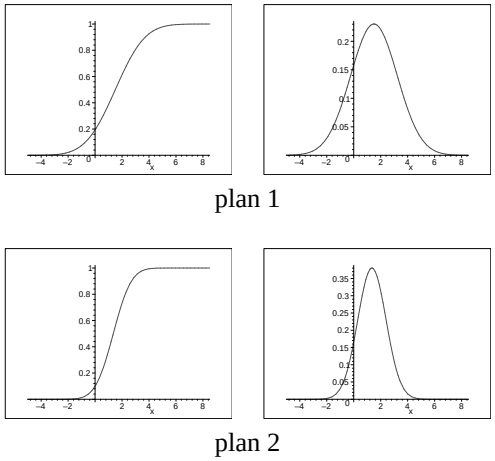


Figure 2: Probability density and distribution of the energy consumption of the two generated plans.

**Comparison with Interval Arithmetic** In an interval arithmetic approach instead, the resource consumption would be represented by intervals which are symmetric to the mean value and cover, for example, 90% of the possible events. With this constraints we would get the interval $[2.36, \ldots, 5.65]$ for `moveAtoB`, $[2.18, \ldots, 6.83]$ for `drill`, and $[4.13, \ldots, 5.17]$ for `milling`.

The consumption of the first plan would then be represented by $[4.54, \ldots, 12.48]$, the consumption of the second plan by $[6.49, \ldots, 10.82]$. It is impossible to make any profound prediction about the success of the execution of the plan this way, because the upper bounds of the intervals of both plans are greater than the initial resource, which is 10.

Another advantage of random variables is the possibility to compute cumulative probabilities. When using interval arithmetic, an interval is computed for every resource. It allows for a (rather weak) conclusion about the respective resource (certain over-allocation, certain non-over-allocation, or no conclusion). The accumulation of an overall conclusion concerning various resources is not supported. When using random variables, we are able to deal with plans that contain multiple uncertain continuous resources: Given $n = |\mathcal{R}|$ resources as part of a domain model, the operators in our model contain up to $n$ random variables to describe their resource consumption. Like in the example above, we first compute the cumulative density $D_i$ for every resource $r_i \in \mathcal{R}$. Secondly, the probability $p_{r_i} = Pr[\mathcal{Y}^{r_i}_{D_i} \leq 0]$ of an over-allocation is determined as shown above.

In the last step the cumulative probability of the whole plan can be computed. This covers also the probability of a plan failure due to over-allocation. For this computation we assume that all resources $r_i \in \mathcal{R}$ are independent. The probability for a successful execution of the plan $P$ is defined as $p_{\text{plan}} := \prod_i (1 - p_{r_i})$.

## 4 Knowledge-based Planning and Heuristics

In contrast to disjunctive planners (e.g., partial order planners), planners like those based on the *Hierarchical Task Network* (HTN) paradigm make extensive use of domain knowledge (see e.g. (Wilkins and desJardins 2001)). In general, it is these *knowledge-based* planners that are used in real-world applications.

HTN planning (Erol 1995) relies on so-called *tasks* and *task networks*. Roughly, tasks correspond to operators, while task networks correspond to plans. Tasks $t$ are either *abstract* or *primitive* and task networks represent partially ordered (abstract) plans. In our setting, primitive tasks are the STRIPS-based operators introduced in Sections 2 and 3. Abstract tasks are "abstract operators", which can be stepwise refined into task networks using so-called methods. A *method* $m = (t, tn)$ indicates that $t$ can be *decomposed* into the (abstract) plan $tn$. Starting from an initial abstract task, HTN planning performs the stepwise decomposition of abstract tasks until a network is reached that contains only primitive tasks. This network represents a plan solving the initial task. (For further details see also (Biundo and Schattenberg 2001)).

Decomposition methods are part of the domain model. By representing legal decompositions of tasks, methods are a means to encode additional domain knowledge. This knowledge can be used to generate heuristics, which help to speed up planning (Clement *et al.* 2001). In view of continuous resource consumption, we aim at heuristics that allow to find plans with the highest probability for a successful execution. So far, information about resource consumption is only provided with primitive tasks, however. This information has to be propagated along the abstraction hierarchy in order to enable the derivation of useful heuristics without enforcing the generation of every possible primitive plan beforehand.

In this section we show how to extract heuristics for monotonic continuous uncertain resource consumption out of an HTN domain model. Monotonic resource consumption means that the domain model contains only operators that either consume the resource or leave it untouched.

One problem with generating heuristic values for resource consumption represented through normal-distributed random variables is that we have to deal with two values $\mu$ and $\sigma^2$ which cannot be minimized independently from each other. Moreover, as shown in the example in Section 3, it is not appropriate to reduce only the mean value $\mu$ of a consumption because a large variance can also cause a lower probability for a successful plan execution (Daniels and Carrillo 1997). The first question is how to represent an underestimation $\mathcal{Y}^r$ of uncertain continuous resource consumptions $\mathcal{X}^r$. We propose to use a random variable for this purpose too. We choose as $\mathcal{Y}^r$ a variable that has the lowest mean value and the lowest variance of all distributions $\mathcal{X}^r$. This is expressed using the $\min$ function:

$$\min\left({}_{o_i}\mathcal{X}^r_{\mathcal{N}(\mu_i,\sigma_i)}\right) := \mathcal{Y}^r_{\mathcal{N}\left(\max_i \mu_i, \min_i \sigma_i\right)}$$

$\min({}_{o_i}\mathcal{X}^r)$ is an underestimation of all ${}_{o_i}\mathcal{X}^r$. In a second step, abstract tasks have to be enriched by random variables that represent these heuristic values. To this end, every task is extended by one random variable for every uncertain continuous resource that is consumed by an operator that appears in any decomposition of this task.

The algorithm that generates the heuristic values works bottom-up. All primitive tasks are associated with the random variables of the respective operator definition. The algorithm searches all task networks that contain only tasks which are already associated with random variables. For each such task network the consumption for every resource is computed as it is done for a plan (cf. Section 3). As we restrict ourselves to monotonic resource consumption, the order in which the primitive tasks are executed can be neglected. The consumption is stored in a random variable,

which is propagated to all tasks that are related to the task network by a method.

After all task networks $tn_1, .., tn_n$ into which a task $t$ can be decomposed are processed, the underestimation for $t$ is computed by $\min({}_{tn_1}\mathcal{X}^r, \ldots, {}_{tn_n}\mathcal{X}^r)$, where ${}_{tn_i}\mathcal{X}^r$ is a propagated value for a resource in task network $tn_i$. This is illustrated in Figure 3 where the abstract task $t$ can be decomposed using one of the methods $m_1, ..., m_3$. The pro-

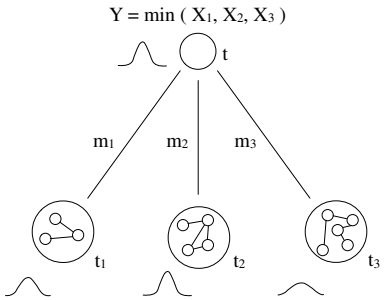


Figure 3: An example of the upward propagation

cess continues until a heuristic value has been computed for each task. Since our domain model is supposed to contain only non-recursive methods, termination of the algorithm is obvious.

In order to compute its total consumption, each task network is processed once. This means, the complexity is $O(m)$, where $m$ is the number of tasks in the largest task network. Recall, that the computation of the convolution is linear, when using normal-distributed random variables. For each task the heuristic value is computed using the $\min({}_{t_1}\mathcal{X}^r, \ldots, {}_{t_n}\mathcal{X}^r)$ function. This means $2 \cdot n$ comparisons have to be carried out to find the minimal mean value and variance. So the complexity of the algorithm is $O(t \cdot m + a \cdot b)$, where $t$ is the number of task networks, $m$ is the number of tasks in the largest task network, $a$ is the number of abstract tasks, and $b$ is the maximum number of methods for a task. This computation has to be done for every resource separately, thus the complexity for creating heuristic values for all $\rho$ resources in a domain is $\rho \cdot O(t \cdot m + a \cdot b) = O(\rho \cdot t \cdot m + \rho \cdot a \cdot b)$.

The propagation algorithm can be used this way to extract underestimations for continuous resource consumptions for each abstract task. It provides the underestimations as heuristic values in a domain analysis step prior to planning. During the planning process the heuristic values can be used to prune the search space by rejecting decompositions that may lead to resource consumptions that are too large or too uncertain.

## 5 Related Work

$\beta$-robust scheduling for single machines is presented in (Daniels and Carrillo 1997), where the total flow time of all scheduled jobs is minimized. In this context, information is gathered about the execution time of single tasks and the duration of the abstract action is estimated by a maximum likelihood, the result of which is a random variable. A fast heuristic function for scheduling performance is compared with correct but slow computations, and it is shown how to select the schedule which promises the best performance.

In a NASA challenge paper (Bresina *et al.* 2002), a Mars rover is used as an example to describe the problem of uncertain continuous variables. There is also given an overview of previous work on planning under uncertainty and the issues are discussed why the discrete probability model of those planners is generally not applicable to the presented rover problem.

The Graphplan planning system has been extended to handle contingencies with probability distributions that are used to represent continuous uncertain resource consumption (Dearden *et al.* 2002). In contrast to common planning problem descriptions, the planning problem has a set of goals with usability annotations. The planner tries to maximize the usability of the whole plan by satisfying a subset of the given goal set. To achieve this, the extended planning algorithm first generates a *seed* plan which is incrementally extended by contingency branches to reduce the probability of failure and increase the usability of the plan.

A lot of work is done in the field of handling *discrete* probabilities in planning, of which we address that about epsilon-safe planning here (Goldman and Boddy 1994). It deals with the feature of uncertain sensing actions and introduces an approach to generate an $\varepsilon$-safe plan, which means to generate a plan that has only a probability of $\varepsilon$ to fail in execution.

Our idea on heuristic propagation of resource consumption is loosely based on (Clement *et al.* 2001), an approach in which intervals on abstract tasks are used as a heuristics function for resource consumption on the action layer. But uncertain resources consumption in operators is not discussed.

## 6 Conclusion and Discussion

We have described an approach to handle uncertainty w.r.t. continuous resource consumption in AI planning. Resources are represented by continuous normal-distributed random variables. By adopting appropriate stochastic concepts, the consumption probabilities of multiple resources can be accumulated. With that, overall probabilities for the successful execution of aggregate plans can be computed. The approach extends to hierarchical planning in a straightforward way. We have shown how it can be embedded into an HTN-based planning framework, where it allows for the derivation of heuristics to guide the planning process towards solutions that meet certain probability thresholds w.r.t. the consumption of critical resources.

So far, our approach is restricted to monotonic resource

consumption and the use of non-recursive HTN methods. In the following, we will briefly sketch how these restrictions can be overcome.

The restriction to non-monotonic resource consumption can easily be abandoned. The heuristic values can be computed in exactly the same way as presented for monotonic resource consumption. However, the heuristic is much weaker then. This is because the order in which resources are produced and consumed by tasks plays an important role. To investigate this in detail, experiments are required. The results are expected to provide detailed information about the cases in which the heuristic still works and cases for which new heuristics have to be developed.

To extend the approach to recursive HTN methods is somewhat harder. The reason is that the generation of heuristic values can only be done if invariants for all recursive methods (resp. the corresponding tasks networks) can be provided. However, an automated generation of such invariants is not feasible. The question therefore is: Can recursive methods be restricted in a way such that invariants can be generated without restricting the expressiveness to much? Alternatively, the user has to annotate the invariants by hand.

The approach presented here belongs to the class of conformant planners. It does not use sensors or conditional branches. To extend our approach to conditional planning, a conditional HTN planner is required. Another possibility would be to transfer the heuristic generation method to a respective planning approach. In the case of continuous resource consumption, the events possibly influencing the execution of a plan are infinite, however. Therefore, it is necessary to branch for intervals or just some quartile of the random variable like it is discussed in (Bresina *et al.* 2002). The problem with splitting a random variable is to find an appropriate density for the resulting intervals.

The approach we have presented in this paper will be implemented in the context of a hybrid planning architecture that integrates HTN and partial order causal link planning (Biundo and Schattenberg 2001).

## References


Piergiorgio Bertoli, Alessandro Cimatti, Marco Roveri, and Paolo Traverso. Planning in nondeterministic domains under partial observability via symbolic model checking. In Bernhard Nebel, editor, *Proc. of the 17th Intern. Joint Conf. on AI (IJCAI-01)*, pages 473–478. Morgan Kaufmann, 2001.

Susanne Biundo and Bernd Schattenberg. From abstract crisis to concrete relief – A preliminary report on combining state abstraction and HTN planning. In Amedeo Cesta and Daniel Borrajo, editors, *Proc. of the 6th European Conf. on Planning (ECP-01)*, LNCS, pages 157–168. Springer Verlag, 2001.

J. Bresina, R. Dearden, N. Meuleau, S. Ramakrishnan, D. Smith, and R. Washington. Planning under continuous time and resource uncertainty: A challenge for AI. In Adnan Darwiche and Nir Friedman, editors, *UAI '02, Proc. of the 18th Conf. in Uncertainty in AI*, pages 77–84. Morgan Kaufmann, 2002.

Bradley J. Clement, Anthony C. Barrett, Gregg R. Rabideau, and Edmund H. Durfee. Using abstraction in planning and scheduling. In Amedeo Cesta and Daniel Borrajo, editors, *Proc. of the 6th European Conf. on Planning (ECP-01)*, LNCS, pages 145–156. Springer Verlag, 2001.

Richard Daniels and Janice Carrillo. $\beta$-robust scheduling for single-machine systems with uncertain processing times. *IIE Transactions*, 29(11):977–985, 1997.

Richard Dearden, Nicolas Meuleau, Sailesh Ramakrishnan, David E. Smith, and Rich Washington. Incremental contingency planning. In *Proceedings of the ICAPS-03 Workshop on Planning under Uncertainty and Incomplete Information*, pages 38–47, 2002.

Kutluhan Erol. *Hierarchical Task Network Planning: Formalization, Analysis, and Implementation*. PhD thesis, University of Maryland, 1995.

Robert P. Goldman and Mark S. Boddy. Epsilon-safe planning. In Ramon López de Mántaras and David Poole, editors, *UAI '94: Proc. of the Tenth Annual Conf. on Uncertainty in AI*, pages 253–261. Morgan Kaufmann, 1994.

Stephen M. Majercik and Michael L. Littman. MAXPLAN: A new approach to probabilistic planning. In R. Simmons, M. Veloso, and S. Smith, editors, *Proc. of the 4th Intern. Conf. on AI Planning Systems (AIPS'98)*, pages 86–93. AAAI Press, Menlo Park, CA, 1998.

M. Peot and D. Smith. Conditional nonlinear planning. In James Hendler, editor, *Proc. of the 1st Intern. Conf. on AI Planning Systems (AIPS'92)*, pages 189–197. Morgan Kaufmann, 1992.

Daniel S. Weld, Corin R. Anderson, and David E. Smith. Extending Graphplan to handle uncertainty and sensing actions. In Charles Rich and Jack Mostow, editors, *Poceedings of the 15th National Conf. of AI (AAAI-98)*, pages 897–904. AAAI Press, Menlo Park, CA, 1998.

D.E. Wilkins and M. desJardins. A call for knowledge-based planning. *AI Magazine*, 22(1):99–115, 2001.