Formalizing Resources for Planning

Tania Bedrax-Weiss, Conor McGann and **Sailesh Ramakrishnan**
QSS Group, Inc.
NASA Ames Research Center
Moffett Field, CA 94035
{tania,cmcgann,sailesh}@email.arc.nasa.gov

Abstract

In this paper we present a classification scheme which circumscribes a large class of resources found in the real world. Building on the work of others we also define key properties of resources that allow formal expression of the proposed classification. Furthermore, operations that change the state of a resource are formalized. Together, properties and operations go a long way in formalizing the representation and reasoning aspects of resources for planning.

Introduction

Historically, allocation of tasks to resources has been considered part of the scheduling problem, and largely omitted from the planning literature. In recent times, the importance of such resource allocation decisions in planning has been recognized (Smith, Frank, & Jónsson 2000; Long *et al.* 2000). Though progress has been made to meet this challenge (Laborie 2001), the state of the art has not yet advanced to a point where we have a comprehensive treatment of resources as an inherent part of a planning framework. More specifically, efforts to characterize the types of resources of interest have been incomplete and largely constrained by the availability of efficient algorithms to reason with them. Furthermore, approaches to natively incorporate resources into domain descriptions have been largely absent.

We believe that the absence of explicit types of resources

- obfuscates the semantics of the model,
- impedes detection of domain modeling errors,
- complicates the mapping to efficient implementations that could be tailored to particular resource types, and
- hinders domain analysis.

For example, resources have not yet been incorporated explicitly in the PDDL 2.1 specification (Fox & Long 2003) although they can be represented through the use of functional expressions. This is illustrated in Figure 1 which describes an action *fly* consuming a resource *fuel*. In the action definition, the fuel consumption is expressed as an effect decreasing the level of *fuel*. Fuel is not identified explicitly as a resource. Furthermore, the inherent properties of *fuel* and the way in which it is allowed to change are not represented.

In this paper we present a classification scheme that will circumscribe a large class of resource types found in the real world. We use and build upon the large body of work both in the planning and scheduling communities. We first develop, through exploration of real world examples, a description of resources from a planning perspective. We define a set of properties that characterize a resource. We then present a classification scheme based on these properties. We go on to explain how this classification will identify transactions on resources. We present examples throughout to illustrate the terms introduced. Where possible, we follow the PDDL 2.1 syntax in the hope that it will be familiar to the reader. We then review related work in PDDL and other languages, in terms of their methods and limitations to address the needs outlined. We conclude with a brief synopsis and discussion of future work.

```
(:durative−action fly
   :parameters (?p − plane ?t − traveller ?a ?b − location)
   :duration (= ?duration (flight−time ?a ?b))
   :condtion (and (at start (at ?p ?a))
              (at start (at ?t ?a))
  (over all (inflight ?p))
  (over all (aboard ?t ?p))
  (at start  (>= (fuel−level ?p) (* (flight−time ?a
   ?b) (consumption−rate ?p)))))
   :effect (and (at start (not (at ?p ?a)))
             (at start (not (at ?t ?a)))
             (at end (at ?p ?b))
             (at end (at ?t ?b))
 (at start (inflight ?p))
             (at end (not (inflight ?p)))
             (at end (not (aboard ?t ?p)))
             (at end (decrease (fuel−level ?p) (* (flight−time ?a
   ?b) (consumption−rate ?p))))))
```

Figure 1: Example of an activity on a resource in PDDL 2.1

Classification

Classification by Example

The following examples are designed to illustrate the various features of resources that are of interest to the modeler.

Consider the cargo bay of the space shuttle. Many items of different sizes are placed in the bay and consume volume.

The space is used when an item is placed in it. It is made available again when the item is removed. We can consider the space as an example of a reusable resource. In contrast consider the fuel in a fuel tank. Once consumed it is destroyed permanently. Fuel, in this case, is an example of a consumable resource. If the fuel tank can be refueled then it is an example of a replenishable resource. Process by-products which are never used are examples of producible resources.

The battery on a planetary rover is an example of a continuous resource. Within the capacity of the battery any amount of energy can be drawn at a time. In contrast disk space on a hard drive is consumed in discrete chunks (bytes). This is an example of a discrete resource.

A printer is an example of a single-capacity resource since it prints only one job at a time. On the other hand, a passenger aircraft contains numerous seats representing a multi-capacity resource.

A fuel container typically has a fixed volume, and therefore a fixed capacity. Alternatively, a battery whose capacity degrades over time is an example of a variable capacity resource.

Seats on an airplane are also examples of a deterministic resource because the state of the resource is known precisely. The energy of a battery is an example of a stochastic resource because of the inherent uncertainty in the amount of the resource available.

The fuel tank in a car is an example of an exclusive resource because refueling is not allowed while the engine is running. Data bandwidth is an example of a shared resource because multiple activities can use the bandwidth simultaneously.

A cargo bay has specific restrictions for both weight and volume. Loading a cargo bay consumes both weight and volume at different rates. Weight and volume are two distinct dimensions of the same resource so this is an example of a multi-dimensional resource (Smith & Becker 1997).

Keeping these examples in mind, we proceed to define properties that precisely categorize resources.

Resource Properties

In this section we present a set of properties that can be used to describe a quantitative resource i.e. a resource with capacity and availability described in terms of numeric quantities. We assume all measures of quantity are represented by a generic unit rather than actual units of measure. We also assume that all conversion operations are defined and occur outside of the resource definitions and operations. Furthermore, we assume that all quantities are greater than zero.

We adopt the following notation for domain definitions. Let $\mathbb{R}$ be to the domain of real numbers; let $\mathbb{N}$ be the domain of natural numbers;[1] let $\mathbb{Z}$ be the domain of whole numbers; let $\mathbb{T}$ be the domain of time. Notice that we explicitly defer commitment to whether time is represented by natural numbers or real numbers, leaving the choice to implementation. Furthermore, let $\Re$ be the universe of resources and let $\mathbb{X}$ be the universe of transactions.[2] Let $[t_1, t_2] = \{x \in \mathbb{T} \mid t_1 \leq x \leq t_2\}$ be an interval of time. Let t equal to $[t, t]$ by definition.

The following properties are defined for a resource.

1. *Level*: The amount of available resource. The level is defined as $\mathcal{L}{:}\Re \times \mathbb{T} \rightarrow [\mathcal{L}_{min}, \mathcal{L}_{max}]$ where $\mathcal{L}_{min}, \mathcal{L}_{max} \in \mathbb{R}$ for continuous resources and $\mathcal{L}_{min}, \mathcal{L}_{max} \in \mathbb{Z}$ for discrete resources. We assume that plans may have temporal flexibility so the level is given by an interval.
2. *Level Limit*: The instantaneous physical limit of available resource. The level limit is given by an upper and a lower bound and is defined as $\mathcal{LL}{:}\Re \times \mathbb{T} \rightarrow \langle \mathcal{LL}_{min}, \mathcal{LL}_{max} \rangle$ where $\mathcal{LL}_{min}, \mathcal{LL}_{max} \in \mathbb{R}$ for continuous resources and $\mathcal{LL}_{min}, \mathcal{LL}_{max} \in \mathbb{Z}$ for discrete resources.
3. *Rate*: The change in level per unit of time. Formally, $\mathcal{R}{:}\Re \times \mathbb{T} \rightarrow [\mathcal{R}_{min}, \mathcal{R}_{max}]$ where $\mathcal{R}_{min}, \mathcal{R}_{max} \in \mathbb{R}$ for continuous resources and $\mathcal{R}_{min}, \mathcal{R}_{max} \in \mathbb{Z}$ for discrete resources. Because plans may have temporal flexibility, the level is given by an interval.
4. *Rate Limit*: The limit imposed on the rate. The rate limit is given by a lower and an upper bound and is defined as $\mathcal{RL}{:}\Re \times \mathbb{T} \rightarrow \langle \mathcal{RL}_{min}, \mathcal{RL}_{max} \rangle$ where $\mathcal{RL}_{min}, \mathcal{RL}_{max} \in \mathbb{R}$ for continuous resources and $\mathcal{RL}_{min}, \mathcal{RL}_{max} \in \mathbb{Z}$ for discrete resources.
5. *Transactions*: The set of all transactions that may or must occur by a given instant of time, defined by $\mathcal{T}{:}\Re \times \mathbb{T} \rightarrow \mathbb{X}$
6. *Completed Transactions*: The set of all transactions that must occur by a given instant of time, defined by $\mathcal{CT}{:}\Re \times \mathbb{T} \rightarrow \mathbb{X}$
7. *Pending Transactions*: The set of all transactions that may overlap a given instant. $\mathcal{PT}{:}\Re \times \mathbb{T} \rightarrow \mathbb{X}$
8. *Transaction Count*: The number of total transactions at an instant of time, defined by $\mathcal{TC}{:}\Re \times \mathbb{T} \rightarrow [\mathcal{TC}_{min}, \mathcal{TC}_{max}]$ where $\mathcal{T}_{min}, \mathcal{T}_{max} \in \mathbb{N}$. Because plans may have temporal flexibility, the transaction count is given by an interval.
9. *Transaction Limit*: The limit imposed on the number of concurrent transactions. The transaction limit is defined as $\mathcal{TL}{:}\Re \times \mathbb{T} \rightarrow \langle \mathcal{TL}_{min}, \mathcal{TL}_{max} \rangle$ where $\mathcal{T}_{min}, \mathcal{T}_{max} \in \mathbb{N}$. Because a resource may have upper and lower limits on the transaction count, the transaction limit is given by an interval.
10. *Horizon*: The time interval over which the resource can process transactions and answer queries. Formally, $\mathcal{H} = [\mathcal{H}_s, \mathcal{H}_e]$, where $\mathcal{H}_s, \mathcal{H}_e \in \mathbb{T}$.

Variations of properties 1 through 9 can be defined to capture consumption and production on the resource, e.g. production rate, production level, consumption transaction count, consumption level limit, consumer transactions. It should be noted that the levels, rates, counts and transaction sets are derived values based on the occurrence of transactions on the resource. In contrast, limits are imposed directly to capture constraints on the state. It should also be

[1] We assume that $\mathbb{N}$ includes the number zero.

[2] Transactions will be described in more detail in the next section.

noted that limits are not used in the definitions of the properties they intend to constrain. This is deliberate, since no commitment is given regarding the enforcement policies of constraints in these definitions.

In addition to the formal definitions provided for each property, certain relationships can be observed among them for any given instant of time.

1. The maximum transaction count is equal to the sum of the maximum producing transaction count and the maximum consuming transaction count. $\mathcal{TC}_{max} = \mathcal{TC}_{prod,max} + \mathcal{TC}_{cons,max}$
2. Similarly, the minimum transaction count is equal to the sum of the minimum producing transaction count and the minimum consuming transaction count. $\mathcal{TC}_{min} = \mathcal{TC}_{prod,min} + \mathcal{TC}_{cons,min}$
3. The maximum rate of change of the resource is the difference between the maximum production rate and the minimum consumption rate. $\mathcal{R}_{max} = \mathcal{R}_{prod,max} - \mathcal{R}_{cons,min}$
4. The minimum rate of change of the resource is the difference between the minimum production rate and the maximum consumption rate. $\mathcal{R}_{min} = \mathcal{R}_{prod,min} - \mathcal{R}_{cons,max}$
5. The set of completed transactions is a subset of the set of all transactions. $\mathcal{CT} \subseteq \mathcal{T}$
6. The set of pending transactions is a subset of the set of all transactions. $\mathcal{PT} \subseteq \mathcal{T}$
7. No transaction can belong to both the pending transactions and the completed transactions. $\mathcal{PT} \cap \mathcal{CT} = \emptyset$
8. A transaction is either a pending or a completed transaction. $\mathcal{CT} \cup \mathcal{PT} = \mathcal{T}$
9. The maximum resource level at an instant of time is a function of the quantities of the completed production transactions, the completed consumption transactions, and the quantities of some subset of the pending transactions. The subset is chosen so as to maximize the the overall resource level. $\mathcal{L}_{max} = \sum_i qp_i(t) - \sum_i qc_i(t) + \sum_i qx_i(t)$,
   where
   $$\begin{aligned} qp_i(t) &: \exists\, \texttt{produce}(r, qp_i(t)) \in \mathcal{T}_{prod} \cap \mathcal{CT}, \\ qc_i(t) &: \exists\, \texttt{consume}(r, qc_i(t)) \in \mathcal{T}_{cons} \cap \mathcal{CT}, \\ qx_i(t) &: \exists\, \texttt{produce}(r, qx_i(t)) \in S \text{ or} \\ &\quad\ \exists\, \texttt{consume}(r, qx_i(t)) \in S \end{aligned}$$
   where $S \subseteq \mathcal{PT}$ is chosen so as to maximize the sum.
10. The maximum resource level at an instant of time is a function of the quantities of the completed production transactions, the completed consumption transactions, and the quantities of some subset of the pending transactions. The subset is chosen so as to minimize the the overall resource level. $\mathcal{L}_{min} = \sum_i qp_i(t) - \sum_i qc_i(t) - \sum_i qx_i(t)$,
    where
    $$\begin{aligned} qp_i(t) &: \exists\, \texttt{produce}(r, qp_i(t)) \in \mathcal{T}_{prod} \cap \mathcal{CT}, \\ qc_i(t) &: \exists\, \texttt{consume}(r, qc_i(t)) \in \mathcal{T}_{cons} \cap \mathcal{CT}, \\ qx_i(t) &: \exists\, \texttt{produce}(r, qx_i(t)) \in S \text{ or} \\ &\quad\ \exists\, \texttt{consume}(r, qx_i(t)) \in S \end{aligned}$$
    where $S \subseteq \mathcal{PT}$ is chosen so as to maximize the sum of the $qx_i(t)$.

```
(:resource fuel−in−a−tank
    :level−limit [0 2000]                 ;all tanks in this domain
                                          ;can hold atmost 2000 units
    :production−rate−limit [0 100]        ;for all tanks in the domain
                                          ;atmost 100 units can be
                                          ;pumped in per unit time
    :consumption−rate−limit [0 100]
    :num−transactions−limit [0 4]         ;atmost 4 transactions can be
                                          ;done on this resource.
)

(fuel−in−a−tank fueltank1
    :level−limit (                        ;represents a limit profile
          (over [0 6] [0 1000])           ;where upto 6 am and after
          (over [6 18] [0 2000])          ;6 pm the tank can hold 1000
          (over [18 24] [0 1000])         ;and between 6am−6pm hold
          )              ;.               ;2000

    :production−rate−limit [0 42.5]
    :consumption−rate−limit [0 75.5]
    :num−transactions−limit [0 3]
    :level 876.34                         ;in the init state, this
                                          ;particular tank has 876 units
                                          ;in it.
    :num−producers−limit [0 2]            ;this tank has two inlet
                                          ;valves
    :horizon [9 17]                       ;fuel can be drawn only during
                                          ;normal working hours.
    )
```

Figure 2: Example in pseudo-PDDL describing some properties of a resource

$\mathcal{L}_{max}$ and $\mathcal{L}_{min}$ represent what is informally called the "resource envelope". Different algorithms (Laborie 2001; Muscettola 2002) compute the envelope with varying degrees of accuracy. The degree of accuracy depends on how carefully the set S is chosen.

There are other distinctions that can be expressed in terms of the properties described above. For example, persistence and expiration dates can be described in terms of the horizon. Similarly, scheduled unavailability can be described in terms of limits on the number of transactions or the rates of change. Other researchers (Powell, Shapiro, & Simao 2001) have mentioned concepts such as active and passive and have characterized numbers of simultaneous producers and consumers. These concepts can be expressed in terms of the properties defined above.

Figure 2 presents an example illustrating the use of these properties in specifying a resource. Restrictions included in the definition of the resource *fuel-in-tank* restrict the properties of all instances of that resource. Restrictions in the definition of the instance *fuel-tank1* impose further restrictions for that particular instance alone. Additionally, the level-limit of *fuel-tank1* is expressed as a profile over time. In this case, the profile is piecewise constant i.e. between the hours of 6 am and 6 pm the maximum level-limit and the minimum level-limit are constants (2000 and 0 respectively).

Resource Categories

The set of resource categories informally introduced earlier provide a means to qualitatively describe the nature of a resource. The set of properties introduced earlier provide a means to quantitatively describe the nature of a resource. In this section we formalize the former in terms of the latter.

A resource can be categorized based on how it may be produced or consumed.

1. *Consumable*: A consumable resource is a resource that is decreased by some activities but is not produced by any activities in the system. For example, a resource such as ammunition may be depleted by firing a weapon. In a mode where the plan of attack prohibits resupply then additional ammunition may not be obtained. This means that $\mathcal{L}$ is monotonically non-increasing.
2. *Producible*: A producible resource is a resource that is created by some activities but is not consumed by any activities in the system. A waste-product of an industrial process may be an example of this. This means that $\mathcal{L}$ is monotonically non-decreasing.
3. *Replenishable*: A replenishable resource can be both produced and consumed as part of the same system. Any ordering of production and consumption transactions are allowed on the resource, e.g battery power which may be produced if it is in the charger, as well as consumed, if the device it powers is turned on.
4. *Reusable*: A reusable resource is a replenishable resource that is produced and consumed with the additional constraint that producing and consuming transactions must happen in tandem. That is, for any interval of time, two consecutive consumption or production transactions are not allowed.

We can further describe resources as discrete or continuous based on the quantities produced or consumed.

1. *Discrete*: The resource is consumed, produced, or used in discrete quantities. For example, disk space is allocated in chunks of bytes. A printer consumes single sheets of paper from the paper bin. If a resource is discrete, the levels, limits and rates are all discrete.
2. *Continuous*: The resource is consumed, produced, or used in continuous quantities, e.g energy stored in a battery. If a resource is continuous, the levels, limits, and rates are all continuous.

Depending on the amount (divisibility or unit) we can distinguish between single and multiple capacity resources.

1. *Single Capacity*: The resource can be thought of as one unit which must be consumed as a whole. This characteristic implies a restriction on the level limit such that
$$\mathcal{LL} = \begin{cases} 0 & \text{if it is being consumed} \\ 1 & \text{otherwise} \end{cases}$$
2. *Multiple Capacity*: The resource represents multiple units which can be used or consumed by different operations. This characteristic implies a restriction on the level limit such that $\mathcal{LL}_{max} > 1$ and $\mathcal{LL}_{min} \neq \mathcal{LL}_{max}$.

The variation of capacity over time allows us to distinguish between fixed and variable capacity resources.

1. *Fixed capacity*: The level limit of a resource is fixed over time, e.g. a gas tank has a fixed capacity to store gasoline. This characteristic imposes a constraint on the level limit such that the level limit is constant with respect to time.
2. *Variable capacity*: The level limit of a resource is a function of time, e.g. a battery whose capacity degrades over time. This characteristic agrees with our definition of level limit.

The level of certainty with which one can determine the capacity allows us to classify resources as deterministic or stochastic.

1. *Deterministic capacity*: The capacity is precisely determined. This characteristic agrees with our definition of level limit.
2. *Stochastic capacity*: The capacity is determined probabilistically. This characteristic is only mentioned for completeness since we provide no formal treatment of this characteristic in this paper.

If simultaneous transactions are allowed to operate on resources then the resource is shared as opposed to exclusive.

1. *Shared*: Using the resource at less than full capacity allows others to use the resource simultaneously, e.g. a battery on a rover typically provides energy simultaneously to a number of devices. This characteristic imposes a constraint on the transaction limit such that $\mathcal{TL}_{max} > 1$ for all time instants.
2. *Exclusive*: Only one activity is allowed to access the resource at a time, regardless of the amount of resource used, e.g. a restaurant table that seats 10 but only 6 people are seated at the table. Even though it seats 10, the remaining 4 seats are made unavailable. This characteristic imposes a constraint on transactions such that $\mathcal{TL} = \langle 0, 1 \rangle$ for all time instants.

There are two additional characteristics noted in (Smith & Becker 1997) that are also important: single-dimensional vs. multi-dimensional, and pooled vs. not pooled. A resource is multi-dimensional if it has more than one aspect to its quantity and each aspect must be updated together but at different rates. A cargo bay with constraints on weight and volume is an example of a multi-dimensional resource. Each *load* activity would simultaneously consume both weight and volume in different quantities. A resource may be aggregated into a resource pool. This allows the domain model to abstract out details of individual resources while preserving the individual nature of each resource for final allocation. We have not had the time to further explore these concepts. For the purpose of this paper we consider all resources to be single-dimensional and omit treatment of resource pools.

Figure 3 gives an example of three resources which have been defined in terms of the categories described above. Qualitative categories impose restrictions on quantitative properties. Note that as before, properties may be additionally restricted.

```
;A camera that can be used to take a picture at a time
(:resource camera
   :categories (exclusive
      fixed
      single−capacity
      discrete
      deterministic
      reusable))


;Earth communication window with a fixed bandwith
(:resource Earth−Communication
   :categories (shared
      fixed
      multi−capacity
      continuous
      deterministic
      reusable)
   :horizon ((540 550) (1020 1030) )

;Solid state data storage disk on which data can be written and erased
;after downloading.
(:resource data−storage−disk
   :categories (exclusive
      fixed
      multi−capacity
      discrete
      deterministic
      replenishable))
```

Figure 3: Examples in pseudo-PDDL describing resources based on their categories

Operations on Resources

Resource categories provide a means to qualitatively describe the nature of a resource. Resource properties, on the other hand, provide a means to quantitatively describe both its nature and its current state. Given this model, the operations to effect state change (transactions) can be formally defined. Throughout, we take the view that all operations on resources should be context free, i.e. we make no assumptions about the activity context that is causing the transaction.

Resource Transactions

Resource transactions are caused by actions in a plan to change the state of a resource. Transactions on resources have fixed semantics that are fully defined based on each transaction type. We assume that transactions can always be applied and that if no other transactions occur in the period (instant or interval) of time over which they're applied, the effect is always observed.[3]

The term *quantity* is used in transaction specifications to indicate the amount of the given resource to be transacted. In the general case, quantity may vary over time. In this case, quantity is a function of time returning a number, an element of $\mathbb{R}$ for continuous resources or an element of $\mathbb{N}$ for discrete resources. The precise semantics of the *quantity* function, however, is implementation specific and we define it externally. Computing with these functions has to be well-defined, so we require that all *quantity* functions be commutative and convolvable. It is also likely we could enforce additional restrictions given the particulars of a resource description and more explicit insight into the semantics of a *quantity* function. At this time, no scheme has been developed to formalize the semantics of these functions to allow such checks for correctness by analysis of the model.

We define the following transactions.

1. *Consume*: Consumption can occur either at an instant of time or over an interval of time.
   - $\texttt{at}\ t\ \texttt{consume}(r, q)$ If a consume transaction and no other transactions are operating on a resource at the specified time, the transaction has the effect of reducing the level by the specified quantity at that instant of time. This transaction also increases the transaction count and consumer transaction count by one.
   - $\texttt{over}\ [t_s, t_e]\ \texttt{consume}(r, q(t))$ If a consume transaction and no other transactions are operating on a resource over the specified time interval, the transaction has the effect of reducing the level by the quantity evaluated at each instant of time in the interval. This transaction also increases the transaction count and consumer transaction count by one for all $t_s \leq t \leq t_e$.

   A *consume* transaction is only allowed on resources which are *replenishable* or *consumable*.
2. *Produce*:
   - $\texttt{at}\ t\ \texttt{produce}(r, q)$ If a produce transaction and no other transactions are operating on a resource at the specified time, the transaction has the effect of reducing the level by the specified quantity at that instant of time. This transaction also increases the transaction count and producer transaction count by one.
   - $\texttt{over}\ [t_s, t_e]\ \texttt{produce}(r, q(t))$ If a produce transaction and no other transactions are operating on a resource over the specified time interval, the transaction has the effect of reducing the level by the quantity evaluated at each instant of time in the interval. This transaction also increases the transaction count and producer transaction count by one for all $t_s \leq t \leq t_e$.

   A *produce* transaction is only allowed on resources which are *replenishable* or *producible*.
3. *Use*: A use operation is defined in terms of consume and produce transactions such that a consume transaction occurs at the start of the use operation and a produce transaction occurs at its end. The quantity consumed is equal to the quantity produced. This operation increases the total number of transactions by two. $\texttt{over}\ [t_s, t_e]\ \texttt{use}(r, q)$ which evaluates to
   $\texttt{at}\ t_s\ \texttt{consume}(r, q)$ and
   $\texttt{at}\ t_e\ \texttt{produce}(r, q)$.
   A *use* transaction is only allowed on *reusable* resources and is allowed to use only a fixed quantity.

[3] If there is any concurrency of transactions it is not always possible to say that if a production of 5 units occurs, the level := level + 5.

Queries and Constraints on resources

Once a resource is defined in terms of its properties, planners can access state of the resource during planning by querying these properties at any instant of time. It is possible to gain access to how a resource changes over time through complex queries that perform arithmetic calculations of the properties over time. The information provided by these queries is very useful in guiding a planner in its search to find a resource compliant plan. Specifically, queries about excess level of resource and the number of transactions provide a planner with the means to look ahead in the planning process and backtrack if necessary. For example, the planner can attempt to balance the consumption of the resource by prefering time intervals where there is the excess capacity is large.

The limit properties of a resource already constrain the operations that can be performed on a resource. For example, the level-limits constrain the total amount of production and consumption, while the transaction count limits constrain the number of activities producing or consuming a resource. More significantly, using the ability to build complex queries over properties, additional constraints can be incorporated into the action definitions (in conditions and effects). Since the properties are defined over time, it is also possible to define constraints that hold only over specific time periods. Complex and dynamic domains can be more naturally expressed through these constraints.

Related Work

Our work is focused on the development of a scheme for classifying and reasoning with resources in planning and we confine ourselves to a discussion of work related to this effort. For a broader examination of the issues of incorporating resources into planning systems see (Long *et al.* 2000). For a discussion of algorithmic issues more pertinent to implementation see (Laborie 2001; Muscettola 2002; Srivastava 2000; Zhou & Smith 2002).

Our work shares some common ground with other efforts to formalize representations and reasoning with resources (Smith & Becker 1997; Yang & Geunes 2001). We differ primarily in our extension to cover epistemological issues, and also in the details of the classfication.

IxTeT (Laborie & Ghallab 1995), ILOG (Laborie 2001) and O-Plan2 (Tate, Drabble, & Kirby 1994) provide examples of planning frameworks that have incorporated complex resource handling capabilities. The scope of their representation is, not surprisingly, constrained by the functionality available in their systems. They each have similar expressive power to handle resources that can be variations of the cross-product of {single-capacity, multi-capacity} and {reusable, consumable, replenishable}. They do not separate the concepts of shareability and capacity, nor do they support the notion of requiring a resource without diminishing its level. No support is provided for resource pools, though approximations are proposed which provide aggregation by creating a new resource of aggregated capacity, e.g. a pool of 10 people becomes a resource of capacity 10. Though they do provide explicit type support for resources, the type structures defined do not account for the variety of characteristics identified in this paper.

More recently, a number of planners have been developed which exploit resource constraints and resource state to control search (Do & Kambhampati 2002; Koehler 1998; Haslum & Geffner 2001; Kvarnstrom & Magnusson 2002). Our work draws from these efforts in developing commonly used terms, transactions, queries and constraints on resources.

Deficiencies in PDDL 2.1 (Fox & Long 2003) with respect to dealing with resources have been documented (Kvarnstrom & Magnusson 2002; Frank, Golden, & Jonsson 2003). For example, treatment of resources implicitly through numeric variables excludes access to aspects of resources that could be useful as operator preconditions, constraints or control rules. Furthermore, the necessity to reference resources as preconditions and effects in this form imposes an artificial serialization of activities. We further believe that the absence of explicit type support for resources makes models more difficult to comprehend due to the natural semantics of resources being implicit. It also makes it more error-prone since one cannot check compatibility of operations on the resource, e.g. producing a consumable resource. Furthermore, it makes it more difficult to map to efficient implementations that could be tailored to particular resource types.

Figure 4 is a representation of the example in figure 1 using our richer representation. A key feature is the representation of the continuous consumption of fuel over an interval of time in the effect of the action *fly*. The semantic of the consume statement is to evaluate the consumption rate (an external function that is time dependent) at various instants of time (determined by the planner implementation) to determine the quantity to be consumed. This quantity is then used in the consume operation of the resource to suitably change its level.

Conclusion

In this paper we have presented a classification scheme which circumscribes a large class of resources found in the real world. Building on the work of others we have also defined key properties of resources that allow formal expression of the proposed classification. Furthermore, operations which alter, query and constrain resource state have been presented. Together this work forms a classification and semantic description that goes a long way in formalizing the representation and reasoning aspects of resources for planning.

Due to the limits of time, a number of outstanding issues have been deferred. For example, no formal treatment is given to resource pools i.e. the ability to aggregate distinct heterogeneous resources into a pool which can be treated as a single resource and still allow allocation of transactions to individual members. Nor have the semantics for multi-dimensional resources been formalized, e.g. simultaneous treatment of the weight and volume aspects of a cargo bay. It may be worthwhile to consider explicitly modeling these composite semantics and provide operations that reflect this.

The properties of resources clearly support expression of temporally scoped limits. This seems adequate to represent

```
(:resource fueltank–in–a–airplane
  :properties (exclusive fixed multi–capacity continuous deterministic
  consumable)
)

(:durative–action fly
   :parameters (?p – plane ?t – traveller ?a ?b – location)
   :duration (= ?duration (flight–time ?a ?b))
   :condtion (and (at start (at ?p ?a))
              (at start (at ?t ?a))
  (at start  (>= (level (fuel–tank ?p) (* (flight–time ?a
   ?b) (max–consumption–rate ?p))))))
   :effect (and (at start (not (at ?p ?a)))
            (at start (not (at ?t ?a)))
            (at end (at ?p ?b))
            (at end (at ?t ?b))
            (over [start end] (inflight ?p))
            (at end (not (inflight ?p)))
            (at end (not (aboard ?t ?p)))
            (over [start end] (consume  (fuel–tank ?p)
   (consumption–rate ?p)))))

(:objects fueltank1 plane1 city1 city2)
(:init(plane plane1)
(airport city1)
(airport city2)
(fueltank–in–a–airplane fueltank1
         :level–limit [0 2134.5]
         :level  872.7
         :consumption–rate–limit [0 10.6])
     (= (fuel–tank plane1) fueltank1))

(:extern  (#t (consumption–rate ?p)))
```

Figure 4: Examples in pseudo-PDDL describing resources and their use in an action

important patterns such as shift rotations or recurring communication windows. However, no methods have been defined to concisely impose cyclic or recurring patterns of constraints on resource properties. This will be investigated in the context of integration of resources into a formal modeling language.

A key issue for reasoning with resources is balancing the costs of checking resource constraints with the benefits that may be obtained to inform search. In scheduling, the set of activities is fixed. In planning, new activities are introduced throughout the process. Consequently, the challenge of deriving useful information from resource propagation is much greater in planning than in scheduling. This issue is highlighted by reported degradations of planner performance (Srivastava & Kambhampati 1999) as more resources are added to the problem, which is counter-intuitive. Approaches to address this have studied deferment of resource reasoning until all activities have been selected (Srivastava 2000), which effectively recasts resource reasoning as a scheduling problem addressed after the planning phase. An important extension of the work described in this paper is to explore techniques which exploit the rich semantics of the proposed representation. More sophisticated queries may be utilized to inform search. More powerful constraints may be specified to make stronger inferences despite the fact that all activities have not been selected. More refined, explicit characterizations of resources and transactions may allow specialization of resource handling algorithms based on model descriptions.

Throughout this paper it has been assumed that all resources are deterministic. It is an open question how the formalism described in this paper could or should be extended to reflect uncertainty. Furthermore, it is assumed that the semantics of external functions describing transaction quantities are defined outside of the model. Although some progress has been made in dealing with this issue, i.e. by requiring such functions to be convolvable and commutative as a prerequisite for admissibility, it is preferable to describe and enforce the semantics of such functions explicitly and unambiguously. It seems plausible that one could require registration of external functions with formal signatures. One could further introduce specified type mechanisms to characterize different classes of functions. The means to achieve this more generally is an open question.

With a view to practical application, and in support of further experimentation, we plan to incorporate the formalism proposed, together with any extensions we may develop, into formal modeling languages for domain, problem and heuristic descriptions. We also plan to integrate the proposed formalism to a planning framework such as that described in (Frank & Jónsson 2003). This effort is likely to raise many interesting issues and trade-offs in terms of modeling fidelity and computational complexity. Furthermore, domain analysis techniques shall be investigated to aid model specification and allow domain-independent specialization of implementations. This work would require mapping data structure and algorithm choices to particular classes of resources.

Acknowledgements

The authors would like to acknowledge the input of Jeremy Frank, Ari Jónsson and Paul Morris, who collaborated in early discussions preceding this work. We also wish to acknowledge the support of the NASA Intelligent Systems Program.

References

Do, M. B., and Kambhampati, S. 2002. Sapa: A scalable multi-objective heuristic metric temporal planner. *Journal of Artificial Intelligence Research*. To appear.

Fox, M., and Long, D. 2003. PDDL2.1: An extension of PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research*. To appear.

Frank, J., and Jónsson, A. 2003. Constraint-based attribute interval planning. *Constraints Journal, Special Issue on Planning*. To appear.

Frank, J.; Golden, K.; and Jonsson, A. 2003. The loyal opposition comments on plan domain description langauges. Sumitted to ICAPS Workshop on PDDL.

Haslum, P., and Geffner, H. 2001. Heuristic planning with time and resources. In *IJCAI-01 Workshop on Planning with Resources*.

Koehler, J. 1998. Planning under resource constraints. In *European Conference on Artificial Intelligence*, 489–493.

Kvarnstrom, J., and Magnusson, M. 2002. TAL planner in the third international planning competition: Extensions and control rules. *Journal of Artificial Intelligence Research*. To appear.

Laborie, P., and Ghallab, M. 1995. Planning with sharable resource constraints. In *14th International Joint Conference on Artificial Intelligence*, 1643–1649.

Laborie, P. 2001. Algorithms for propagating resource constraints in ai: Existing approaches and new results. In *IJCAI-01 Workshop on Planning with Resources*.

Long, D.; Fox, M.; Sebastia, L.; and Coddington, A. 2000. An examination of resources in planning. Technical report, Department of Computer Science, University of Durham.

Muscettola, N. 2002. Computing the resource envelope for stepwise constant resource allocations. In *Principles and Practice of Constraint Programming*, 139–152.

Powell, W. B.; Shapiro, J. A.; and Simao, H. P. 2001. A representational paradigm for dynamic resource transformation problems. Technical Report CL-01-05, Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08544.

Smith, S. F., and Becker, M. 1997. An ontology for constructing scheduling systems. In *1997 AAAI Symposium on Ontological Engineering*. AAAI Press.

Smith, D.; Frank, J.; and Jónsson, A. 2000. Bridging the gap between planning and scheduling. In *Knowledge Engineering Review, 15(1):61–94, 2000.* Cambridge University Press.

Srivastava, B., and Kambhampati, S. 1999. Scaling up planning by teasing out resource scheduling. In *ECP*.

Srivastava, B. 2000. Realplan: Decoupling causal and resource reasoning in planning. In *AAAI/IAAI*, 812–818.

Tate, A.; Drabble, B.; and Kirby, R. 1994. O-plan2: An architecture for command and control. In Fox, M., and Zweben, M., eds., *Intelligent Scheduling*. San Mateo, California: Morgan Kaufmann.

Yang, B., and Geunes, J. 2001. Resource-constrained project scheduling: Past work and new directions. Technical Report 2001-6, Department of Industrial and Systems Engineering, University of Florida.

Zhou, Q., and Smith, S. F. 2002. An efficient consumable resource representation for scheduling. In *3rd International NASA Workshop on Planning and Scheduling for Space Applications*.