Multimedia Adaptation Decisions Modelled as Non-Deterministic Operations*

Fernando López[1], Dietmar Jannach[2], José M. Martínez[1], Christian Timmerer[2], Hermann Hellwagner[2], Narciso García[3]

[1]GTI, EPS – Universidad Autónoma de Madrid, {f.lopez,josem.martinez}@uam.es

[2] AINF, ITEC – Klagenfurt University, Austria, {firstname.lastname}@uni-klu.ac.at

[3]GTI, ETSIT – Universidad Politécnica de Madrid, narciso@gti.ssr.upm.es

Abstract

This paper describes how a multimedia adaptation framework can automatically decide the sequence of operations to be executed in order to adapt an MPEG-21 Digital Item to the MPEG-21 description of the usage environment in which it will be consumed. The main innovation of this work with respect to previous multimedia adaptation decision models is that in the proposed approach decisions can be made without knowing the exact behaviour of the operations that are going to be executed.

1. Introduction and state of the art

Multimedia adaptation decision taking allows the automatic configuration of an adaptation engine in order to produce content that can be directly consumed in a given usage environment. KoMMa [1] and CAIN [2] are both multimedia adaptation frameworks capable of performing this sort of automatic self-configuration. Additionally, in both engines the inputs and outputs are compliant to the MPEG-21 standard [3], a framework that greatly simplifies the integration within large-scale multimedia systems that are aware of those open interfaces.

In general, multimedia automatic decision-taking systems like [1] and [2] are based on the assumption that the behaviour of the operations to execute is well known before taking decisions about which operations to choose and in which order to execute them. In our work, however, we are proposing a multimedia adaptation engine where third parties may integrate adaptation operations, and as a consequence we cannot always foresee the exact behaviour and, as a result, the output of these operations.

Artificial Intelligence (AI) planning techniques [4] enable an intelligent agent to accomplish automatic decisions before acting. *States* are passive entities that collect the relevant aspects of the problem under consideration, whereas *actions* are active entities that transform the world from one state to another, e.g., by changing one or more features of the world's state [1][4].

Traditional computer science has used the notion of an *operation* (i.e., a procedure, function, or subroutine) to refer to a portion of code that performs a specific task. In computer science the actions that can be executed may be represented as operations where parameters are not only used to designate the input state (as in the case of traditional planning actions) but also include the capacity to choose the output that we desire to obtain after executing the operation.

2. Non-deterministic operations

When the above convention to represent operations is used, we consider it to be very useful to discriminate between *state parameters* that reflect the input state of the world, and *target parameters* that signal the desired output state. For example, let us suppose that the operation *resize*("*movie.mpg*",480,320) modifies the size of "*movie.mpg*" to frame size 480x320 pixels. In this case, "*movie.mpg*" is a state parameter and 480, 320 are target parameters that designate the frame size of the desired output state movie in pixels.

Furthermore, within the standard theory of computation, deterministic algorithms are algorithms where the output state can be precisely determined from the input state. Conversely, non-deterministic algorithms are algorithms with one or more internal choices where several continuations are possible, without a specification of which one will be taken, but where it is guaranteed that all the paths always arrive at a valid solution. Note that the standard theory of computation's non-deterministic algorithms is a different concept to prob-

* Work supported by the European Commission (IST-FP6-027685 – MESH and IST-FP6-038463 – ENTHRONE II), Spanish Government (TEC2007-65400 - SemanticVideo) and Comunidad de Madrid (S-0505/TIC-0223 - ProMultiDis-CM). This work is also supported by the Ministerio de Educación y Ciencia of the Spanish Government through the FPU grant under the name of the first author.

978-0-7695-3130-4/08 $25.00 © 2008 IEEE
DOI 10.1109/WIAMIS.2008.35

ability theory's stochastic process whose behaviour is non-deterministic. Automatic decision systems have implicitly considered operations as *deterministic operations.* In this work, however, we propose to model multimedia conversion operations as *non-deterministic operations*, i.e., where – as a result – the output state cannot be determined from observing the input state parameters and the target parameters. The motivation behind this proposal is that third party multimedia conversion may accomplish several continuations without a precise specification of which one will be taken.

In the literature, the term *state* has been used to refer to both the predictable result that one knows will be obtained when a *decision is taken* about the operations (without executing them) and the result that will be obtained after *executing* an operation. This is due to the assumption that there exist only deterministic operations, and under that condition both results have the same values. However, the existence of non-deterministic operations makes necessary to discriminate between potential states and realized states. Specifically, we propose to use the term *potential state* to refer to the feasible values of the state that a non-deterministic operation may produce and we propose to use the term *realized state* to refer to the resulting state obtained after executing a non-deterministic operation. Hereafter, we will use the term *decision phase* to refer to an algorithm that, given a potential input state and an operation, provides the potential output state. Conversely, we will use the term *execution phase* to refer to an algorithm that, given a realized input state (that represents the real world), executes an operation with the realized input state as parameter and produces another realized output state that corresponds to the result of the execution. Also, according to this taxonomy, it is important to note that realized states correspond to the subset of the potential states that are achieved when the operation is actually executed. In the simple case of deterministic operations, potential state values are entirely defined and both states (potential states and realized states) are the same. We propose to use the term *unique potential state* to refer to this last class of states. However, the introduction of non-deterministic operations requires us to introduce what we refer to as *multiple potential states*, i.e., states that cannot be totally anticipated because they are the output of a non-deterministic operation.

Traditional preconditions, postconditions, and invariants have been represented with first order predicates that have shown to be a successful way to model the states. In this work, we are going to evaluate an alternative representation, i.e., we propose to represent unique potential states and realized states by a *set of variables*, where each *variable* is a label with a unique associated value (e.g., *width* = 320). Note that variables are suitable to represent the input and output of deterministic operations, because given an input state and a deterministic operation it is always possible to determine the values of the variables of the output state. However, as a non-deterministic operation must produce a valid solution, we can presume that it is possible to represent the postconditions using ranges or a set of values. This observation motivates our proposal to represent multiple potential states by a *set of properties*, where each *property* is a label along with a set of associated potential values. In this way, properties have the form *of key-values pairs* where the *key* is a label and the *values* correspond to a possibly empty set of homogeneous elements (e.g., *width* = {320, 640, 800, 1024} or *width* = [100..5000]).

Similar to classical planners that search for a sequence of actions that lead from an initial state to a goal state, we propose the use of a *non-deterministic planner*, i.e., an algorithm capable of finding a sequence of non-deterministic operations (or simply referred to as *sequence of operations*) that leads from a potential initial state to a potential goal state. Due to the fact that we have to cope with non-deterministic operations, there might exist several sequences of operations that lead to the goal state. Thus, this algorithm must be capable of finding both the *set of feasible sequences of operations* and the target parameter values that must be provided to each non-deterministic operation in order to follow a specific sequence of operations. Note that the non-deterministic planner that we propose to address the multimedia adaptation decision problem does not correspond to classical conditional planning [4], because conditional planning introduces the assumption of contingency decisions as the plan execution progresses and usually removes the existence of linearly ordered sequences of actions. On the other hand, the non-deterministic planner that we propose to use corresponds to a sort of planning under uncertainty [4]. In particular, "planning under uncertainty" often deals with states that are not totally observable. However, in the proposed non-deterministic planner the states are always totally observable. Besides, planning under uncertainty usually associates probabilities to the outcomes of the actions (e.g., the probability of an error during its execution). In our proposed model, operations correspond to non-deterministic algorithms and we assume that they always lead to valid solutions and, thus, the associated probabilities are totally irrelevant. The main concepts of this algorithm are provided in the next section.

3. Multimedia conversions modelled as non-deterministic operations

Within CAIN, a CAT (Component Adaptation Tool) is a pluggable software module capable of performing several *conversions* to adapt (multi-)media resources, which are represented as *Component* elements of an MPEG-21 DI (Digital Item) [3]. Additionally, the conversion capabilities of a CAT are expressed using the CAT Capabilities [5]. The DM (Decision Module) is the module in charge of choosing the sequence of conversions which adapts an MPEG-21 *Component* [3] to a given MPEG-21 UED (Usage Environment Description) [3]. Subsequently, the EM (Execution Module) will execute the chosen sequence of conversions.

Representation issues. Figure 1 shows the elements that take part in a non-deterministic multimedia conversion.

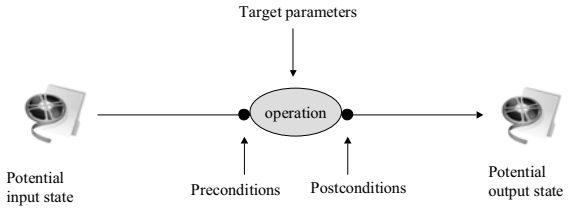

Figure 1: Representation of a non-deterministic conversion.

Traditional planning algorithms make use of first order logic predicates to represent states and actions. We have replaced predicates by a set of properties (e.g., *media_format*, *width*, *height*). In particular, the values of the properties of the potential states result from resolving a set of XPath[1] expressions over the MPEG-21 DI that conveys the *Component* to adapt. The potential goal state is obtained from resolving another set of XPath expressions over the MPEG-21 UED (terminal, network, and user preferences). The values of the properties of the preconditions and postconditions of the conversions result from resolving a set of XPath expressions over the CAT Capabilities. The target parameters are also represented by properties. However, the non-deterministic planner dynamically calculates the values of these properties instead of extracting those values from the existing descriptions.

The non-deterministic planner. As shown in Figure 2, the goal of the non-deterministic planner – implemented within the DM – is to search for a set of feasible sequences of conversions that adapt an MPEG-21 *Component* according to the requirements described in the MPEG-21 UED. In our proposal, there must exist only one UED, and as a consequence, only one potential goal state. There might however exist several variations of the *Component*, i.e., several potential initial states, which is impossible in [1] and [2]. In particular, the initial states are unique potential states, because the value of each property is unique, whereas the goal state is a (possibly multiple-valued) *potential state* since the UED may accept several values in each property (e.g., the terminal accepts several screen sizes). We could think about modelling the initial *Component* and its variations as one multiple potential state that encompasses all the variations. In practice, however, this is not always possible because different variations (represented as *Component* elements) may have different media resources and different *Resource* element descriptions that correspond with different properties (e.g., different coding formats, bitrates, etc.). For these reasons, we decided to use a single potential state for each variation.

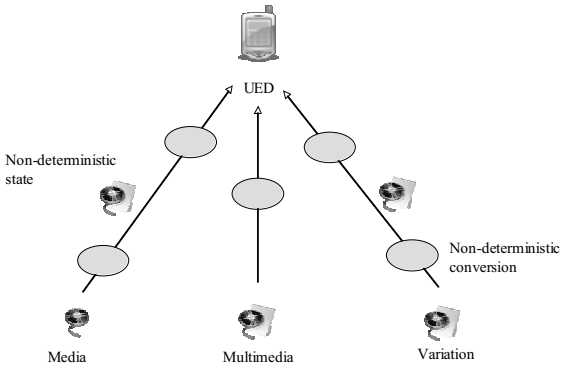

Figure 2: Conceptual view of the non-deterministic planner.

In this context, a *tree of conversions* (ellipses in Figure 2) represents the set of feasible sequences of conversions that we can execute to adapt the media to the UED (i.e., root of the tree representing the goal state).

In summary, the algorithm that searches for a non-deterministic plan is similar to an algorithm that searches for a deterministic plan. The main difference is that potential states are not uniquely defined but may include multiple values[2].

Semantics of the conversion descriptions. We propose the following semantics for the preconditions, postconditions, and invariants of a conversion:

- When a property appears in a precondition of a conversion, this configuration means that the conversion *requires* this property in the input state. Specifically, the input state property must be a subset of the precondition's property values.
- When a property appears in a postcondition of a conversion, this situation must be interpreted in the sense that the conversion *produces* the property. Note that the conversion may create the property (if

[1] http://www.w3.org/TR/xpath

[2] Due to space constraints, use cases and further details of this algorithm have been published at the following URL http://www-gti.ii.uam.es/publications/mad_ndo/

it does not exist in the input state) or modifies the property (if it exists in the input state).

Dealing with incompleteness. It is important to note that there may exist a large number of properties. Therefore, it does not seem reasonable to supply all these properties in every adaptation problem. The CAT implementer may for instance not wish to provide the value of all these properties. On the other hand, the DI and usage environment may arrive at the adaptation engine without values for all these properties. With this observation in mind, we designed a solution capable of dealing with the semantics of incomplete descriptors, which was implemented as follows:

- When an input state does have a property that appears in the preconditions of a conversion, the conversion *cannot receive* the media because the conversion requests a property that is unknown (e.g., when a conversion requests as precondition greyscale images but the input image colour features are not annotated).

To deal with incomplete conversion descriptions, we designed the following solution:

- When a property does not appear in the preconditions of a conversion, this configuration must be interpreted in a sense that – with regard to that property – *every value is acceptable*.
- When a property appears neither in the preconditions nor in the postconditions of a conversion, this situation must be interpreted in a way that the conversion *preserves* the value of this property, i.e., the output state inherits the property of the input state without changes.
- When a property appears in the preconditions of the conversion, but does appear neither in the postconditions nor in the invariants of a conversion, this situation must be interpreted in a way that the conversion *neglects* the value of this output property and – as a consequence – we cannot make any assumption about the value of this property in the output state. Note that "neglects" must not be interpreted in the sense that the CAT necessarily "loses" this property, but in the sense that the CAT says nothing about what is going to happen with this property. If for example a conversion declares the maximum frame rate that it accepts, but does not declare the maximum frame rate that it produces, this situation can arise based on the fact that the media resource produced by the conversion does not have a frame rate at all (e.g., it is an audio resource), or due to the fact that – although the conversion produces video – it does not specify the output frame rate.

Note that the *preserves* solution forces the planner to assume that whenever a property does not appear in the conversion description, the property is not modified by the conversion. Therefore, the CAT must avoid modifying those properties. Note that this is a risky assumption as the CAT implementer is assumed to be careful and precise in his/her design. The other option would be to force the CAT implementer to annotate all the properties that the conversion does not modify which would however become tedious for the CAT implementer. Alternatively, the algorithm that searches for the plan would have to discard all the conversions not properly annotated.

4. Conclusions

In this paper, we demonstrated the convenience of representing multimedia conversions as non-deterministic operations (in contrast to previous multimedia adaptation decision systems [1][2]) and we developed a method to implement a multimedia adaptation engine that automatically decides on the adaptation to perform in order to adapt an MPEG-21 DI to the MPEG-21 UED. Specifically, we argue that this method allows for the integration of third party software modules whose behaviour cannot be completely known. Although the method has only been evaluated with a still small number of multimedia adaptation scenarios, we claim that non-deterministic operations can be easily extended to other sorts of decision-taking problems.

In addition our approach is capable of dealing with absent properties, which we perceive as very useful in practical environments. If interpreted with the proposed semantics (and in conjunction with multi-valued properties), the non-deterministic planner that we propose allows searching in a set of potential states which are only partially determined and which thus significantly reduces the number of states that must be evaluated.

5. References

[1] D. Jannach, K. Leopold, C. Timmerer, H. Hellwagner, "A Knowledge-Based Framework for Multimedia Adaptation", *Applied Intelligence*, 24(2):109-125, 2006.

[2] J.M. Martínez, V. Valdés, J. Bescós, L. Herranz, "Introducing CAIN: A metadata-driven content adaptation manager integrating heterogeneous content adaptation tools", *Proc. of WIAMIS'2005*, Montreux, CH, 2005.

[3] F. Pereira, J.R. Smith, A Vetro (eds.), "Special Section on MPEG-21", *IEEE Trans. on Multimedia*, 7(3), 2005.

[4] S. Russell, P. Norvig. *Artificial Intelligence: A modern approach*, Prentice Hall, 2003.

[5] V. Valdés, J.M. Martínez, "Content Adaptation Capabilities Description Tool for Supporting Extensibility in the CAIN Framework", *LNCS 4105*, Springer Verlag, 2006.