Planning Personalised Museum Visits

Daniel Le Berre and **Pierre Marquis** and **Stéphanie Roussel**
CRIL - CNRS, UMR 8188
Université d'Artois
F-62307 Lens, France

Abstract

In this paper, we consider the problem of designing personalised museum visits. Given a set of preferences and constraints a visitor might express on her visit, the aim is to compute the tour that best matches her requirements. The museum visits problem can be expressed as a planning problem, with cost optimization. We show how to bound the number of steps required to find an optimal solution, via the resolution of an instance of the shortest complete walk problem. We also point out an alternative encoding of the museum visits problem as an optimization problem with pseudo-Boolean constraints and a linear objective function. We have evaluated several constraints solvers, a planner and a tailored solver on a number of benchmarks, representing various instances of the museum visits problem corresponding to real museums. Our empirical results show the feasibility of both the planning and the constraint programming approaches. Optimal solutions can be computed for short visits and "practically good" solutions for much longer visits.

1 Introduction

We present a novel application of planning for the design of personalized museum visits. The need for such personalized visits comes from the fact that for a large number of museums, it is not conceivable to spend time in front of every art piece, especially because of their number which can be huge (exceeding thousands of items in some museums). This is why it makes sense to prepare a visit. To make things more concrete, ask yourself about what you would be doing if you just have one hour and a half to spend at Le Louvre? Our purpose is to help a visitor finding a "good answer" to such a question. While a few museums already offer some pieces of software enabling the user to prepare her visit, only a limited number of predefined tours are considered, and those which best match the user preferences and constraints are pointed out. An example of such a software can be found on Le Louvre website http://www.louvre.fr/en/parcours. An accurate handling of the user preferences and constraints (bearing on the visit duration, the selection of specific art pieces, etc.) is out of reach of such simple approaches, which may output tours which are rather unsatisfactory from the user point of view because they mismatch her preferences in a significant way or do not satisfy by a large amount the expected duration constraint (predefined tours are often several hours long).

We have developed both a planning approach and a constraint programming approach to the design of personalised museum visits. Our approaches start with a number of input pieces of information, including the museum topology, and the user type, preferences and constraints. The museum topology can be modeled as a valued, oriented graph where each room corresponds to a vertex and each "alley" between rooms is an arc. Arcs are valued with durations which depend on the length and the nature of the corresponding "alley" (it can be a corridor, a stair or a lift) and also on the visitor type (individual, group, disabled people, etc.). The graph must be oriented since it can be the case that a flow direction is imposed between some rooms. The museum topology also makes precise the location of every art piece. In our setting, the user preferences are modeled by a utility function supposed to satisfy full additive independence (i.e., the utility of a tour is the sum of the utilities of each art piece considered during the visit). It is obtained via an anytime elicitation process (collaborative filtering is used to suggest default values depending on the visitor type) and results in a mapping associating with each art piece a value (its utility for the visitor). Some constraints on the admissible tours are directly induced from the museum topology: a sequence of rooms is feasible only if there is an "alley" between each pair of successive rooms. Visitors impose some constraints: some of them are automatically triggered from the visitor type, e.g. arcs corresponding to stairs must be ignored for wheel-chaired visitors, other constraints emerge from an interaction with her (expected duration of the tour, inclusion/exclusion of some art pieces, etc.). From all those data, the purpose is to output a tour which respects all the constraints and maximizes the overall utility.

Many recommendation systems have been developed so far to capture the artworks a visitor might be able to appreciate the most (a detailed list of recommendation systems dedicated to museums is given in (Hage et al. 2010)). Among them is the CHIP Art Recommender (Wang et al. 2010) where a visitor profile is built using rated artworks and rated features. A major bottleneck shared by all those recommendation tools is the tour generation issue, which is not always considered or does not guarantee that the visitor preferences are satisfied in the best way. Thus, the tour wizard (Hage et al. 2010) associated to CHIP orders the artworks by decreasing utility order, starting from an empty tour, it tries to add each of them in an iterative way to the current tour when this is possible (i.e., when the constraints are not violated). Clearly enough, such a greedy approach does not offer any optimality guarantee.

The museum visits problem (as informally described above) looks quite similar to some well-known transportation problems considered in operations research (the traveling salesman problem with bounded resources for instance). Indeed, the museum visits problem can be considered as a specific orienteering problem (Vansteenwegen, Souffriau, and Oudheusden 2011), where the objective is to determine visit tours in towns or countries, which maximize the user utility. A detailed list of tools for tourist transportation problems is given in (Kabassi 2010) and one of the most recent tools is described in (Garcia et al. 2010) and (Vansteenwegen et al. 2011). The authors of (Jaén et al. 2011) suggest to reduce the museum visits problem to the orienteering problem approach, so that solving the latter enables solving the former. However, the number of art pieces in a museum can be much larger (by several orders of magnitude) than the number of sites interest in a town so that the solution proposed by (Jaén et al. 2011) does not apply to large museums. The reason is that existing encoding schemes for the orienteering problem require a quadratic number of binary variables in the number of sites. The point is that large museums present thousands of art pieces, so that the corresponding encodings would include millions of variables, making their resolution out of reach of existing solvers, for space reasons.

In the following, the museum visits problem is modeled as a planning problem with cost optimization. We also model it as an optimization problem with pseudo-Boolean constraints and a linear objective function. We note that the corresponding decision problem is NP-complete, which justifies to tackle it using "classical" planning or constraint programming techniques. We model first the museum visits problem as a planning problem with two actions (moving from on room to another one and looking at an artwork), a time constraint and an optimization function. We also point out a specific constraint programming encoding scheme for the museum visits problem, where each art piece corresponds to a Boolean variable; it allows us to escape from the size issue raised up by the encoding scheme suited to the orienteering problem, as discussed above. Our encoding scheme requires the computation of a bound on the number of visit steps, which is performed via the resolution of an instance of a shortest complete walk problem. We prove that the decision problem associated with the shortest complete walk problem NP-complete, so it makes sense to solve it as well using constraint programming techniques.

We have evaluated several state-of-the-art constraint solvers, a planner and a tailored solver on a number of benchmarks, representing various instances of the museum visits problem corresponding to real museums. Our empirical results show the feasibility of both the planning and the constraint programming approaches. Optimal solutions can be computed for short visits using linear programming, and "practically good" solutions for much longer visits look reachable using a hybrid approach combining a constraint programming solver with incremental horizon handling as in planning.

The rest of the paper is organized as follows. In Section 2 we define the museum visits problem in formal terms and identify the complexity of the corresponding decision problem. In Section 3 we describe the methodology followed to solve the museum visits problem; especially, we present both a planning encoding and a constraint programming encoding suited to this problem. In Section 4 we present the benchmarks generated to evaluate our approaches and some empirical results are also pointed out. Finally, in Section 5, we conclude the paper and present some perspectives for further research.

2 The Museum Visits Problem

The museum topology is modeled as a so-called museum graph:

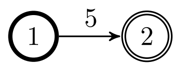
Definition 1 *A* museum graph *is a 5-tuple* $\mathcal{G} = \langle \mathcal{V}, \mathcal{A}, \mathcal{E}, \mathcal{S}, \mathbf{w} \rangle$ *where:*

- $\mathcal{V}$ *is a set of vertices, each vertex representing a room of the museum.*
- $\mathcal{A}$ *is a set of arcs. An arc* $a_{i,j}$ *between two vertices* i *and* j *belongs to* $\mathcal{A}$ *iff it is possible in the museum to go directly from room* i *to room* j *(i.e., without going through a third room). Observe that we do not consider edges but oriented arcs. This is because large museums can impose some flow directions.*
- $\mathbf{w}$ *is a mapping* $\mathbf{w} : \mathcal{A} \to \mathbb{N}^*$ *where* $\mathbf{w}(a_{i,j})$ *represents the duration to go from room* i *to room* j*. We suppose that this duration is strictly greater than 0: going from a room to another one is not instantaneous.*
- $\mathcal{E}$ *is a subset of* $\mathcal{V}$ *representing entrances of the museum.*
- $\mathcal{S}$ *is a subset of* $\mathcal{V}$ *representing exits of the museum.*

It is assumed that $\mathcal{G}$ *has a* complete walk*, i.e., a walk starting at one entrance, finishing at one exit and including all the vertices. Formally, a complete walk is a finite sequence of vertices* $(v_1, v_2, \ldots, v_{n-1}, v_n)$ *of* $\mathcal{V}$ *with* $v_1 \in \mathcal{E}$*,* $v_n \in \mathcal{S}$*, for all* $i \in [\![1; n-1]\!]$*,* $a_{v_i,v_{i+1}} \in \mathcal{A}$ *and for all* $v \in \mathcal{V}$*, there exists* $i \in [\![1; n]\!]$ *such that* $v_i = v$*. In museum terms, this simply means that there exists a tour of the museum which starts at an entrance, finishes at an exit and includes every room of the museum. Obviously enough, this assumption holds for every existing museum.*

A toy example is given in Figure 1(a). The set of vertices is $\mathcal{V} = \{1, 2\}$, the set of arcs is $\mathcal{A} = \{a_{1,2}\}$, the set of entrances is $\mathcal{E} = \{1\}$ (represented by a thick circle), the set of exits is $\mathcal{S} = \{2\}$ (represented by a double circle), the function $\mathbf{w}$ is such that $\mathbf{w}(a_{1,2}) = 5$. Such a graph has a unique complete walk which is $(1, 2)$.

Note that a complete walk can be a non-simple path (a simple path is a walk in which all edges are distinct) hence a non-elementary path (an elementary path is a walk in which all vertices are distinct).



(a) Graph museum $\mathcal{G}$

$\mathcal{P}$	$\mathbf{r}(p_i)$	$\mathbf{u}(p_i)$	$\mathbf{t}(p_i)$
p_1	1	5	3
p_2	1	15	8
p_3	2	11	3

(b) Art pieces

Figure 1: A toy museum

Definition 2 *A* museum visit instance *is a 6-tuple* $\langle \mathcal{G}, \mathcal{P}, \boldsymbol{r}, \boldsymbol{u}, \boldsymbol{t}, \text{TMAX}\rangle$ *where:*

- $\mathcal{G} = \langle \mathcal{V}, \mathcal{A}, \mathcal{E}, \mathcal{S}, \boldsymbol{w}\rangle$ *is a museum graph.*
- $\mathcal{P}$ *is a finite set of art pieces.*
- $\boldsymbol{r}$ *is a mapping* $\mathcal{P} \to \mathcal{V}$. *For each art piece* $p \in \mathcal{P}$, $\boldsymbol{r}(p)$ *is the room where* p *is located.*
- $\boldsymbol{u}$ *is a mapping* $\mathcal{P} \to \mathbb{N}$. *For each art piece* $p \in \mathcal{P}$, $\boldsymbol{u}(p)$ *represents the satisfaction brought by* p *to the visitor. The higher* $\boldsymbol{u}(p)$*, the more satisfied the visitor.*
- $\boldsymbol{t}$ *is a mapping* $\mathcal{P} \to \mathbb{N}$. *For each art piece* $p \in \mathcal{P}$, $\boldsymbol{t}(p)$ *(denoted* t_p*) represents the duration that the visitor would spend by visiting* p.
- TMAX *is a non-negative integer representing the maximum amount of time (in time units) the visitor is willing to spend in the museum.*

An example is given at Figure 1(b).

Definition 3 *A* solution *to a museum visit instance* $\langle \mathcal{G}, \mathcal{P}, \boldsymbol{r}, \boldsymbol{u}, \boldsymbol{t}, \text{TMAX}\rangle$ *is a pair* $sol = \langle walk, O\rangle$ *where* $walk$ *is a sequence* $(v_1, \dots, v_n)$ *of vertices* $\mathcal{V}$ *and* O *a subset of* $\mathcal{P}$ *such that:*

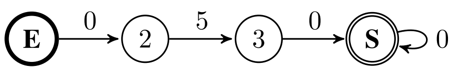
1. $v_1 \in \mathcal{E}$, $v_n \in \mathcal{S}$ *and* $\forall i \in [\![1; n-1]\!]$, $a_{v_i, v_{i+1}} \in \mathcal{A}$
2. *if* $p \in O$ *then* $\boldsymbol{r}(p) \in walk$
3. $\sum_{i=1}^{n-1} \boldsymbol{w}(a_{v_i,v_{i+1}}) + \sum_{p \in O} \boldsymbol{t}(p) \leq \text{TMAX}$

With each $sol = \langle walk, O\rangle$*, is associated* $score(sol) = \sum_{p\in O} \boldsymbol{u}(p)$. *A* best solution *to* $\langle \mathcal{G}, \mathcal{P}, \boldsymbol{r}, \boldsymbol{u}, \boldsymbol{t}, \text{TMAX}\rangle$ *is a maximal solution in terms of score.*

Intuitively, the first condition constrains the walk to start at an entrance and to finish at an exit. The second condition ensures that every art piece of the solution is in a room of the walk. The third condition guarantees that the total time spent in the museum (duration of the walk and duration in front of each art piece of O) does not exceed the limit given by the visitor.

Solving a museum visits problem consists in computing a best solution given a museum visit instance. Given the museum graph of Figure 1(a) and the set of art pieces of Figure 1(b), an museum visit instance is $\mathcal{I} = \langle \mathcal{G}, \{p_1, p_2, p_3\}, \mathbf{r}, \mathbf{u}, \mathbf{t}, 15\rangle$. A solution of this instance is $\langle (1, 2), \{p_1\}\rangle$. The score of this solution is 5. The reader can check that a best solution for the instance $\mathcal{I}$ is $\langle (1, 2), \{p_1, p_3\}\rangle$. No solution has a score higher than 16.

The decision problem MUSEUM associated with the museum visits problem (which is basically an optimization problem) consists in determining whether a solution having a score greater than or equal to u, where u is a given integer, exists, given a museum visit instance. This problem is computationally difficult in the sense that no polynomial-time (deterministic) algorithm exists for it, unless P = NP; indeed, MUSEUM is NP-complete. It is easy to show that MUSEUM remains NP-hard under the restriction when set $\mathcal{V}$ contains a unique room since, in such a case, it can be viewed as (the decision version of) a knapsack problem. The complexity of MUSEUM justifies resorting to planning or constraint programming as an approach suited to the resolution of the museum visits problem.



(a) Graph museum $\mathcal{G}_{\mathbf{E},\mathbf{S}}$

$\mathcal{P}$	$\mathbf{r}_{\mathbf{E},\mathbf{S}}(p_i)$	$\mathbf{u}(p_i)$	$\mathbf{t}(p_i)$
p_1	2	5	3
p_2	2	15	8
p_3	3	11	3

(b) Art pieces

Figure 2: The toy museum, updated

3 Solving the Museum Visits Problem

In order to make the encoding scheme easier to grasp, we add to the museum graph two specific vertices **E** and **S** to $\mathcal{V}$, which respectively represent a virtual entrance and a virtual exit ("sink"). For each entrance e of $\mathcal{E}$, we add an arc $a_{\mathbf{E},e}$ to $\mathcal{A}$ such that $\mathbf{w}(a_{\mathbf{E},e}) = 0$. For each exit s of $\mathcal{S}$, we add an arc $a_{s,\mathbf{S}}$ to $\mathcal{A}$ such that $\mathbf{w}(a_{s,\mathbf{S}}) = 0$. We also add an arc $a_{\mathbf{S},\mathbf{S}}$ to $\mathcal{A}$ such that $\mathbf{w}(a_{\mathbf{S},\mathbf{S}}) = 0$. This allows us to consider a unique entrance and a unique exit. The updated set of arcs and the updated weight function are respectively denoted by $\mathcal{A}_{\mathbf{E},\mathbf{S}}$ and $\mathbf{w}_{\mathbf{E},\mathbf{S}}$. We consider now that solution walks begin at **E** and finish at **S** instead of beginning at any entrance and finishing at any exit. As well, we can re-use the notion of complete walk: a complete walk now begins at **E** and finishes at **S**.

The updated set of vertices, the updated set of arcs and the updated weight function are respectively denoted $\mathcal{V}_{\mathbf{E},\mathbf{S}}$, $\mathcal{A}_{\mathbf{E},\mathbf{S}}$ and $\mathbf{w}_{\mathbf{E},\mathbf{S}}$. The updated graph is denoted $\mathcal{G}_{\mathbf{E},\mathbf{S}}$. The room mapping **r** can also be modified by the introduction of **E** and **S**. The updated mapping is denoted $\mathbf{r}_{\mathbf{E},\mathbf{S}}$.

The update of the toy museum example given at Figure 1 is presented at Figure 2. A complete walk in this museum is now $(\mathbf{E}, 2, 3, \mathbf{S})$. The update of instance $\mathcal{I}$ is $\mathcal{I}_{\mathbf{E},\mathbf{S}} = \langle \mathcal{G}_{\mathbf{E},\mathbf{S}}, \mathcal{P}, \mathbf{r}_{\mathbf{E},\mathbf{S}}, \mathbf{u}, \mathbf{t}, 15\rangle$ and the best solution for this instance is $sol_{\mathbf{E},\mathbf{S}} = \langle (\mathbf{E}, 2, 3, \mathbf{S}), 16\rangle$. Notice that the only difference with the best solution of $\mathcal{I}$ is the presence of **E** and **S** respectively at the beginning and at the end of the solution walk.

3.1 Planning modeling

The museum visits problem can be easily modeled as a planning problem with two actions. The action `move` allows the visitor to move from one room r_1 to another room r_2 and the action `look-artwork` allows the visitor to look at an artwork. In both cases, a precondition is to make sure that the remaining time is sufficient to execute the action, while an effect is to decrease the value of the remaining available time. The remaining time is modeled using numeric expressions as offered by PDDL 2.1 (Fox and Long 2003). The complete PDDL domain is given below.

```
(define (domain museum)
(:requirements :typing :numerical-fluents)
(:types room artwork)
(:predicates (visitor-at ?r - room)
    (inroom ?a - artwork ?r - room)
    (unseen ?a - artwork)
    (connected ?r1 ?r2 - room))
(:functions (total-utility)
    (remaining-time)
    (utility ?a - artwork)
    (time-look ?a - artwork)
    (time-move ?r1 ?r2 - room))
(:action move
 :parameters (?r1 ?r2 - room)
 :precondition (and (visitor-at ?r1)
                (connected ?r1 ?r2)
    (>= remaining-time (time-move ?r1 ?r2)))
 :effect (and (not (visitor-at ?r1))
          (visitor-at ?r2)
          (decrease (remaining-time)
          (time-move ?r1 ?r2) )))
(:action look-artwork
 :parameters (?a - artwork ?r - room)
 :precondition (and (inroom ?a ?r)
                (visitor-at ?r)
                (unseen ?a)
    (>= remaining-time (time-look ?a)))
 :effect (and (not (unseen ?a))
              (increase (total-utility)
              (utility ?a))
   (decrease (remaining-time) (time-look ?a))
)))
```

The example given at Figure 2 can then be described the following way:

```
(define (problem museum)
(:domain museum)
(:objects
 rE r2 r3 rS - room
 a1 a2 a3 - artwork)
(:init
 (= (total-utility) 0)
 (= (remaining-time) 15)
 (= (utility a1) 5)
 (= (utility a2) 15)
 (= (utility a3) 11)
 (= (time-look a1) 3)
 (= (time-look a2) 8)
 (= (time-look a3) 3)
 (= (time-move rE r2) 0)
 (= (time-move r2 r3) 5)
 (= (time-move r3 rS) 0)
 (= (time-move rS rS) 0)
 (connected rE r2)(connected r2 r3)
 (connected r3 rS)(connected rS rS)
 (unseen a1)(unseen a2)(unseen a3)
 (inroom a1 r2)(inroom a2 r2)(inroom a3 r3)
 (visitor-at rE))
(:goal (and (visitor-at rS)))
(:metric maximize (total-utility)))
```

3.2 Constraint programming modeling

Let us now present a constraint programming encoding of the museum visits problem, which associates with each instance a set of pseudo-Boolean constraints and a linear objective function, such that every variable assignment maximizing the function corresponds to a best solution of the input instance. Let K denote the maximum number of move steps allowed for the visitor. In the encoding scheme, we consider three types of binary decision variables:

- x_p for $p \in \mathcal{P}$. $x_p = 1$ if the artwork p belongs to the solution subset of art pieces, $x_p = 0$ otherwise;
- $x_{p,k}$ for $p \in \mathcal{P}$, $k \in [\![1; K]\!]$. $x_{p,k} = 1$ if the artwork p belongs to the solution subset of art pieces and $\mathbf{r}(p)$ is the k^{th} element of the solution walk, i.e., the art piece p is visited at step k, $x_{p,k} = 0$ otherwise;
- $c_{i,j,k}$ for $i, j \in \mathcal{V}^2_{\mathbf{E},\mathbf{S}}$ such that $a_{i,j} \in \mathcal{A}_{\mathbf{E},\mathbf{S}}$ and $k \in [\![1; K]\!]$. $c_{i,j,k} = 1$ if room i and room j are respectively at positions k^{th} and $(k+1)^{th}$ in the solution walk, i.e., the solution walk arc a_{ij} is taken just after step k, $c_{i,j,k} = 0$ otherwise.

$$max \sum_{p \in \mathcal{P}} \mathbf{u}(p).x_p \quad (1)$$

$$\left(\sum_{k=2}^{K-1} \sum_{a_{i,j} \in \mathcal{A}} \mathbf{w}(a_{i,j}).c_{i,j,k} \right) + \sum_{p \in \mathcal{P}} \mathbf{t}(p).x_p \leq \textsc{Tmax} \quad (2)$$

$$\forall k \in [\![1; K]\!], \quad \sum_{a_{i,j} \in \mathcal{A}} c_{i,j,k} = 1 \quad (3)$$

$$\forall a_{i,j} \in \mathcal{A}, \forall k \in [\![1; K-1]\!], \quad c_{i,j,k} \leq \sum_{a_{j,l} \in \mathcal{A}} c_{j,l,k+1} \quad (4)$$

$$\forall k \in [\![2; K]\!], \forall p \in \mathcal{P}, \quad x_{p,k} \leq \sum_{a_{\mathbf{r}(p),j} \in \mathcal{A}} c_{\mathbf{r}(p),j,k} \quad (5)$$

$$\sum_{a_{\mathbf{E},j} \in \mathcal{A}} c_{\mathbf{E},j,1} = 1 \quad (6)$$

$$\sum_{a_{i,\mathbf{S}} \in \mathcal{A}} c_{i,\mathbf{S},K} = 1 \quad (7)$$

$$\forall p \in \mathcal{P}, \ x_p = \sum_{k=2}^{K} x_{p,k} \quad (8)$$

The objective function (1) measures the visitor satisfaction. Constraint (2) ensures that the total amount of time spent in the museum is not larger than $\textsc{Tmax}$. We sum from $k \geq 2$ because $\forall a_{\mathbf{E},j} \in \mathcal{A}, \mathbf{w}(a_{\mathbf{E},j}) = 0$. Constraint (3) represents non-ubiquity, i.e., the fact that the visitor cannot be at two locations at the same time. Constraint (4) guarantees that the visitor comes out from any room she has come in, so that the solution found is a walk in the museum. Constraint (5) ensures that art pieces can be seen only if the visitor is in the corresponding room. Constraint (6) forces the walk to begin in **E** at step 1. Constraint (7) guarantees that the visitor is in **S** just after step K. The last constraint (8) ensures that i) an artwork is only visited once, and ii) an artwork utility is taken into account in the objective function iff the artwork has been visited at some k.

With this encoding, the number of variables equals to $(|\mathcal{P}| + |\mathcal{A}|) \times K$. The number of constraints equals to

$|\mathcal{A}| \times (K-1) + |\mathcal{P}| \times (2K-1) + K + 3$. In real museums, the number of art pieces is usually much larger than the number or alleys or the length of the shortest complete walk. Hence, the size of the encoding mainly is linear in the number of art pieces.

As we will see on the following, the encoding size of a museum visit instance may have a strong impact on the efficiency of its resolution: the smaller, the better. In order to maintain the encoding size as small as possible, it is important to derive a "good" upper bound for K.

3.3 Computing move bounds

A simple, user-dependent, bound on the number of moves to be considered is given by $K_1 = \lfloor \frac{\text{TMAX}}{min-time-move} \rfloor + 2$. For instance, if the maximal time dedicated to the visit is 15 and each move (except those from the virtual entrance and to the virtual exit) lasts at least 2, then any admissible plan cannot contain more than 9 moves (including those from the virtual entrance and to the virtual exit). While this bound proves efficient when TMAX is rather small, it does not prove useful otherwise.

This explains why we also focused on another, more sophisticated and user-independent, bound K_2 and we consider as the actual bound the minimum of the two: $K = min(K_1, K_2)$. The rationale for K_2 is that the number of moves to be considered to reach an optimal plan cannot exceed the minimal number of moves required to visit each room of the museum. This calls for the computation of the length K_2 of a shortest complete walk of the graph museum:

Definition 4 F-SCW *is the following function problem:*

- ***Input**: an updated museum graph $\mathcal{G}_{\mathbf{E},\mathbf{S}}$*
- ***Output**: the length K_2 of a shortest complete walk of $\mathcal{G}_{\mathbf{E},\mathbf{S}}$*

By definition, K_2 equals to the smallest integer u such that there exists a complete walk of length u in $\mathcal{G}_{\mathbf{E},\mathbf{S}}$ and that there does not exist a complete walk of length $u-1$ in $\mathcal{G}_{\mathbf{E},\mathbf{S}}$. Hence, determining K_2 can be achieved by finding out in an iterative way the maximal value u for which a complete walk of length u exists. The corresponding decision problem (determining whether a complete walk of length $\leq u$ exists in a given oriented graph with a unique source and a unique sink) is referred to as SCW (shortest complete walk). As far as we know, SCW has never been addressed in the graph/OR literature.

If u denotes the tested length, we consider $|\mathcal{V}_{\mathbf{E},\mathbf{S}}|.u$ binary variables $b_{v,i}$, $v \in \mathcal{V}_{\mathbf{E},\mathbf{S}}$ and $i \in [\![1;u]\!]$. $b_{v,i}$ equals 1 if room v is visited at step i, 0 otherwise. For readability reasons, we denote $\mathbf{n}$ the mapping $\mathcal{V}_{\mathbf{E},\mathbf{S}} \rightarrow 2^{\mathcal{V}_{\mathbf{E},\mathbf{S}}}$ that associates with each room v of $\mathcal{V}_{\mathbf{E},\mathbf{S}}$ the set $\{n | a_{v,n} \in \mathcal{A}_{\mathbf{E},\mathbf{S}}\}$, i.e., the set of rooms directly accessible from room v.

The problem of determining whether a complete walk of length u exists in the graph consists in determining whether the following set of constraints is satisfiable:

$$\forall v \in \mathcal{V}_{\mathbf{E},\mathbf{S}}, \ \sum_{i=1}^{u} a_{v,i} \geq 1 \qquad (9)$$

$$\forall v \in \mathcal{V}_{\mathbf{E},\mathbf{S}}, \forall i \in [\![1; u-1]\!], \ a_{v,i} \leq \sum_{n \in \mathbf{n}(v)} a_{n,i+1} \qquad (10)$$

$$\forall i \in [\![1; u]\!], \ \sum_{v \in \mathcal{V}_{\mathbf{E},\mathbf{S}}} a_{v,i} = 1 \qquad (11)$$

$$a_{\mathbf{E},1} = 1 \qquad (12)$$

$$a_{\mathbf{S},u} = 1 \qquad (13)$$

Constraint (9) enforces every room to be visited at least once. Constraint (10) ensures the walk connectivity: if a room v is visited at step j then the room visited at step $j+1$ must be directly accessible from v. Constraint (11) enforces that a unique room is visited at each step. Finally, constraints (12) and (13) respectively ensure that the walk begins in $\mathbf{E}$ at step 1 and ends in $\mathbf{S}$ at step u. For a given u, the encoding contains $u.(|\mathcal{V}_{\mathbf{E},\mathbf{S}}| + 1) + 2$ constraints.

That SCW is solved using a constraint programming approach is justified by:

Proposition 1 SCW *is* NP*-complete.*

Proof 1

- SCW *is obviously in* NP*: it can easily be checked in polynomial time that the guessed sequence $(v_1, \ldots, v_n)$ is a complete walk of the input graph satisfying $n \leq u$.*
- SCW *is* NP*-hard. We reduce in polynomial time the problem of deciding whether an oriented graph with a unique source and a unique sink has a Hamiltonian path (i.e., a path starting from the source, finishing at the sink and including every vertex of the graph exactly once) to* SCW*. This Hamiltonian path problem is known as* NP*-hard (Karp 1972). The reduction is obvious: the input graph is unchanged and the value of u is assigned to the number n of vertices in the graph. Indeed, no complete walk of size $< n$ may exist, hence every complete walk of size n (when it exists) is a shortest complete walk. Furthermore, by construction, a complete walk of size n also is a Hamiltonian path of the graph, and the converse also holds.*

The following result shows that F-SCW belongs to the same complexity class as the traveling salesman problem.

Proposition 2 F-SCW *is in* $\mathsf{FP}^{\mathsf{NP}[O(n^2)]}$.

Proof 2 *Let $w = (v_1, \ldots, v_{K_2})$ be a shortest complete walk of the input graph $\mathcal{G}_{\mathbf{E},\mathbf{S}}$. We show that K_2 is at most quadratic in the number of vertices of $\mathcal{G}_{\mathbf{E},\mathbf{S}}$. w may contain two kinds of vertices: multi-occurrent vertices are those vertices appearing more than once in w and mono-occurrent vertices appearing exactly once. If w contains only mono-occurrent vertices, then its length K_2 is bounded by the number of vertices of $\mathcal{G}_{\mathbf{E},\mathbf{S}}$, and the conclusion follows. In the remaining case, let v_j be the last occurrence of a multi-occurrent vertex v in w and let v_i $(1 \leq i < j)$ be the previous occurrence of v in w. If the subpath $(v_{i+1}, \ldots, v_j)$ of w contains only vertices that already occur in the sequence $(v_1, \ldots, v_i)$, then $w = (v_1, \ldots, v_{K_2})$ cannot be a shortest* *complete walk of $\mathcal{G}_{\boldsymbol{E},\boldsymbol{S}}$ since $(v_1, \ldots, v_i, v_{j+1}, \ldots, v_{K_2})$ is a complete walk of $\mathcal{G}_{\boldsymbol{E},\boldsymbol{S}}$ (it contains the same vertices as w) but is shorter than w. Hence there must exist at least one vertex $v^{i,j}$ of the sequence $(v_{i+1}, \ldots, v_j)$ which does not occur in $(v_1, \ldots, v_i)$. Thus, with every pair v_i, v_j of successive multi-occurrent vertex v in w, we can associate a vertex $v^{i,j}$ of $\mathcal{G}_{\boldsymbol{E},\boldsymbol{S}}$ which does not occur in w before rank i. Hence the number of such pairs is bounded by the number of vertices of $\mathcal{G}_{\boldsymbol{E},\boldsymbol{S}}$. Therefore the number of occurrences of any vertex v in w cannot exceed the number of vertices in $\mathcal{G}_{\boldsymbol{E},\boldsymbol{S}}$. This implies that K_2 is at most quadratic in the number of vertices of $\mathcal{G}_{\boldsymbol{E},\boldsymbol{S}}$, so that $|\mathcal{A}_{\boldsymbol{E},\boldsymbol{S}}|^2$ can be used as the initial value of u.*

4 Experimental Results

4.1 Benchmarks design

No benchmarks for the museum visits problem are currently available. The authors of (Jaén et al. 2011) have re-used orienteering problem benchmarks. However, those benchmarks do not represent real museum visit instances (they were not designed in this objective). Furthermore, our modeling needs more information than the ones needed by the orienteering problem. Thus, we have designed several benchmarks to test our encoding schemes.

We have considered two museums: a small one, the "Palais des Beaux Arts de Lille" (Bea 2012), and a much bigger one, the "National Gallery" (Nat 2012) of London. The characteristics of Palais des Beaux Arts are: 53 rooms, 121 alleys, 1 entrance, 1 exit and 486 artworks. The characteristics of National Gallery are: 119 rooms, 304 alleys, 4 entrances, 4 exits and 1783 artworks.

We used those two museums to test our encoding scheme of the shortest complete walk problem. For this problem, the input is the museum graph as presented in Section 2. Starting with the floor plans available on museum websites, we used the dot language specified in (Gansner and North 2000) to represent the museum topology,[1] as shown on Figure 3.

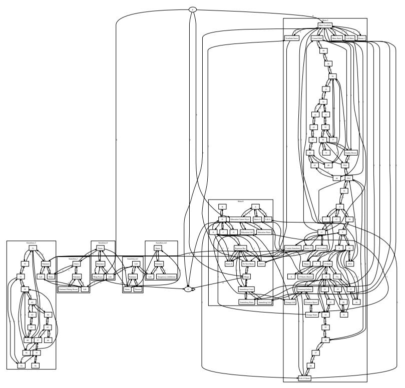


Figure 3: National Gallery Museum topology

Beyond knowing the museum topology, generating museum visit instances requires also to know in which rooms artworks are located. For the Beaux Arts museum, we did not have access to the real museum database but went on-site and estimated the number of artworks in each room. We created the National Gallery artworks database from the information available on the website of this museum. Then, we enriched dot files representing the topology of each museum with the different elements needed for the museum visit instances, i.e., the artworks information, the maximum time bound and the length of the shortest complete walk. Many visitors profiles can be envisioned, associated with many different utility distributions. We considered three distribution types, which differ according to the balance ability they offer. The first one, *unif*, corresponds to users with weak preferences on art pieces (i.e., utility compensations are often feasible). To be more precise, a random integer between 0 and 1000 is associated to each artwork following a uniform distribution. The second utility profile (*fibo*) corresponds to users with mild preferences. We consider 20 utility levels and the utility values for these levels match the first integers of Fibonacci sequence. The artworks number for each of these levels is the same one. Finally, the third profile (*luby*) corresponds to users with marked preferences on art pieces. It is based on a Luby sequence as presented in (Luby, Sinclair, and Zuckerman 1993). The higher the level, the fewer the number of artworks. $\mathbf{u}(p_i) = 2^{k-1}$ if $i = 2^k - 1$, $\mathbf{u}(p_i) = \mathbf{u}(p_{i-2^{k-1}+1})$ otherwise.

In order to generate a wide family of profiles, we fixed the percentage of artworks that do not interest the visitor at all (null) and the percentage of artworks that the visitor considers being the most interesting ones (master). Independently of the function used to define the visitor satisfaction, non-interesting artworks are attributed a null utility and the most interesting artworks are given the maximum utility value. We used the same percentages 0, 10, 20 and 50 for both kinds of artwork, leading to 16 different combinations.

Another parameter to be made precise is the time the visitor is willing to spend in front of each artwork. This time depends on the utility given to an artwork: it seems reasonable to spend more time in front of a very interesting art piece and less time in front of a less interesting one. Nevertheless, this time cannot be null. We consider two cases. In the first case, times to spend in front of artworks and times to go from one room to another are approximately the same. In the second case, times to spend in front of artworks are twice as long as times to go from a room to an adjacent room in the museum. The objective is to observe the impact of different visit rhythms on the behavior of the solvers.

Finally, the complexity of the museum visit instance also depends on the total duration of the visit. Thus, we considered durations of 15, 60, 120, 180 and 300 time units. We generated $480 = |0, 10, 20, 5|^2 \times |0, 10, 20, 50| \times |1.0, 2.0| \times |unif, fibo, luby| \times |15, 60, 120, 180, 300|$ benchmarks for each museum, for a total of 960 benchmarks.

4.2 Empirical results

We used the format of the pseudo-Boolean competition (Manquinho and Roussel 2006) to allow us to try different solvers. We ran the pseudo-Boolean solver SAT4J (Le Berre and Parrain 2010) version 2.3.2 and CLASP (Gebser et al.

[1]Note that we have used subgraphs to model different buildings or different floors of the museum. The visualization has been realized using Graphviz.

2007) 2.1.0 two open source CDCL solvers with native pseudo-Boolean constraints. We also tried CPLEX 12 (CPL 2012) using the PB competition front end to CPLEX written by Vasco Manquinho. Those instances have been generated with steps bound $K = min(K_1, K_2)$ with $K_2 = 67$ for Beaux Arts museum and $K_2 = 192$ for National Gallery.[2] The museum visits instances (without the steps bound) have been encoded as planning instances in PDDL. We ran LPG (Gerevini, Saetti, and Serina 2003) on those instances using the parameters `-n 100 -seed 1234` where n represents the number of plans to compute and *seed* is the number used to initialize the random number generator. The other programs used their default settings. All programs ran on a cluster of bi-Xeon 3.0 GHz processors with 2GB of RAM running Centos. The timeout was set to 900s per instance.

An optimal solution could be computed for 265 out of 960 benchmarks (Table 2), all but one solved by CPLEX. Most of them are "small" instances (TMAX = 15). The pseudo-Boolean solvers were able to provide an answer (not necessarily optimal) for most benchmarks. Such behavior is expected, since CPLEX takes advantage of cutting planes to prove optimality but cannot always provide an answer because it does not explore the Boolean search space. On the other hand, the pseudo-Boolean solvers work on the Boolean search space, guided by the objective function. CLASP solved many more benchmarks than SAT4J, especially on large NG benchmarks. Notably, while LPG did not find any optimal plan, it was able to provide a plan for all benchmarks. We assumed that it was because LPG was looking for solutions with increasing plan length. To check that assumption, we built a dedicated solver called PMV based on SAT4J, because PMV requires a tight integration with the SAT solver and we have a strong expertise with SAT4J; PMV solves the museum visits problem with increasing steps k using a quick timeout, until the bound is reached. We used a timeout of 5 seconds for the first 10 steps, 20 seconds for the following 10 steps and finally 80 seconds per remaining step. That increasing timeout is made to adapt the solver to the increasing search space. The solver also has a timeout for optimizing the solution found, equals to $\frac{1}{5}$ of the global timeout. PMV was also able to provide a solution for all benchmarks, and interestingly outperformed all other approaches.

Figures 4 and 5 reports for each solver under consideration the number of benchmarks for which a solution of given quality (or, dually, of given "regret") has been found. The "regret" of a solution s is assessed by $\frac{u^* - u(s)}{u^*}$, where u^* is the utility of the best solution found. On this figure, values are aggregated so that the "column" OPT (resp. BEST) corresponds to the instances for which no better solution has

Museum	CPLEX	CLASP	SAT4J	LPG	PMV
Encoding	Pseudo Boolean			PDDL	Custom
Number of optimal solutions found (265 out of 960)					
BA	131	88	57	-	-
NG	133	3	3	-	-
Number of solutions found (960 out of 960)					
BA	132	452	408	480	480
NG	134	473	183	480	480

Table 1: Solutions in terms of solver

been found and the solution found has been proved (resp. not proved) optimal, while the "column" 10 (resp. 20, 50 and 50+) corresponds respectively to the number of instances for which the "regret" of the solution found is strictly greater than 0% and less than 10% (resp. strictly greater than 10% and less than 20%, strictly greater than 20% and less than 50%, and greater than 50%).

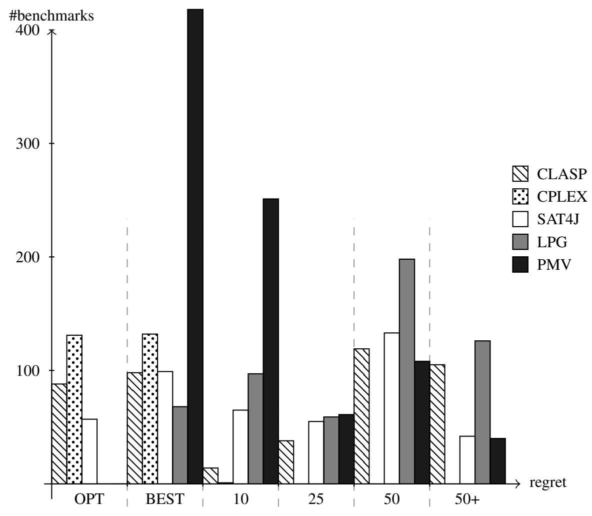


Figure 4: Quality of solutions found on BA museum

In the light of our experiments, finding out optimal tours is not that hard in practice when short-time visits are considered. To be more precise, when TMAX= 15, CPLEX found optimal solutions for all instances (96 BA instances + 96 NG instances). CLASP also exhibited quite good behavior for the BA museum: it found and proved optimal 88 solutions, and found 7 additional optimal solutions without proving them as optimal; however, for the NG museum, its performances decreased significantly: it found and proved optimal 3 solutions, and found 6 additional optimal solutions without proving them as optimal. SAT4J presents a similar performance profile; for the BA museum: it found and proved optimal 57 solutions, and found 7 additional optimal solutions without proving them as optimal; for the NG museum: it found and proved optimal 3 solutions, and found 1 additional optimal solutions without proving it as optimal. LPG found 56 optimal solutions for the BA museum and 10 optimal solutions for the NG museum. Finally, PMV found 82 optimal solutions for the BA museum and 15 optimal solutions for the NG museum. For larger values of TMAX, the number of optimal solutions found decreases: 13 BA and 11

[2]We generated benchmarks for the (decision version of the) shortest complete walk problem and ran CPLEX and CLASP on them using a timeout of 24 hours. We considered 16 step bounds for the BA museum ($k = 65$ to $k = 80$) and 76 for NG ($k = 125$ to $k = 200$). While the value of K_2 could be computed for BA, $\widehat{K}_2 = 192$ is the best upper bound of K_2 we could compute for NG. They confirm that this problem is difficult, especially for large museums.

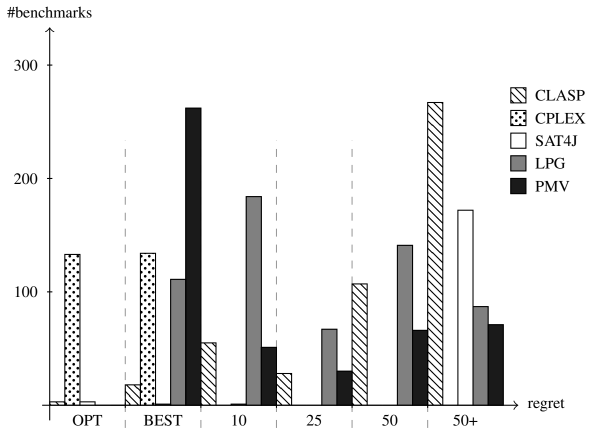


Figure 5: Quality of solutions found on NG museum

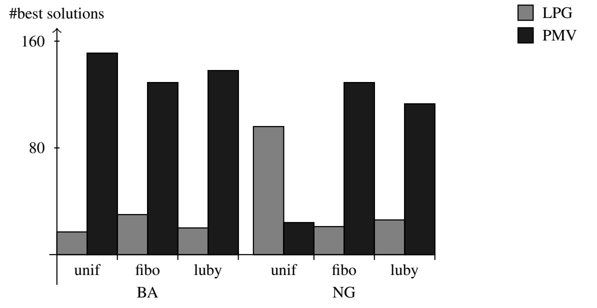


Figure 6: Best solutions depending on the utility profile

NG instances are proven optimal by CPLEX for TMAX=60, 9 BA and 3 NG for TMAX=120, 7 BA for TMAX=180 and finally 3 BA for TMAX=300. However, non optimal solutions can still be computed for those large TMAX.

Globally speaking, PMV showed rather impressive performances compared to the other algorithms when the purpose is to determine "practically good" solutions. On the one hand, the utility value of solutions found by this algorithm are typically very close to the optimal utility value. Of course, this can be measured only for those instances for which optimal solutions can be computed, and they are not so numerous when the time bound is large. On the other hand, whatever the museum, PMV appears by and large as the best performer; indeed, it outputs the best solution found or a solution close to it for a large majority of instances.

It is interesting to note that LPG and PMV behave differently on benchmarks with *unif* profile. Figure 6 depicts the number of best solutions found by LPG and PMV for each utility profile and each museum. We limit ourselves to those solvers because they were able to provide a solution for all benchmarks, unlike the other ones. LPG performs much better than PMV on *unif* profiles for the NG museum. After checking the plans produced by LPG on this profile, we found that LPG was able to produce plans with very few moves. This can be explained by the fact that one room close to an entrance of NG contains many artworks. Since PMV is utility driven and the profile is uniform, it is unlikely that PMV chooses directly artworks in the same room, and such wrong choices cannot be corrected during its limited timeout at a given k.

5 Conclusion

In this paper, we have defined the museum visits problem and modeled it both as a planning problem and a constraint programming problem. Even if the action space is simple, the planning formulation of MUSEUM looks quite natural. The PDDL encoding is much more concise than the constraint-based one we obtained, and as such, it is easier to grasp. In our opinion, the fact that MUSEUM is an optimization problem does not prevent from viewing it as a planning problem as well: from a computational point of view, many planning problems are optimization ones. In some sense, our empirical results suggest that, from the practical side, the right way to consider MUSEUM is both as a planning problem and as a CP one.

Our encoding scheme requires as many Boolean variables as artwork pieces and rooms per step in the museum; it thus improves the encoding scheme which is reached by considering the museum visits problem as an orienteering problem; indeed, this last encoding scheme requires a number of Boolean variables which is quadratic with the number of artwork pieces, which makes it impractical (for space reasons) for many museums. We generated benchmarks derived from two real museums.[3] We experimented with several constraints solvers and a planner on those benchmarks, allowing us to show the feasibility of the approach. The planning approach provides very good results on those problems: LPG was always able to provide a plan within the timeout. Such good results could also be achieved using the constraints encoding via an incremental approach, tailored for this problem. Interestingly, the museum visits problem provides challenging optimization benchmarks for the community. CPLEX seems to be the right choice to get an optimal solution. However, in practice, optimality is not a hard requirement, and using a planner or a pseudo-Boolean solver allows us to retrieve quickly reasonably good solutions.

This work calls for a number of perspectives. Generating more accurate visitor profiles (especially, in terms of utility distributions and time spent in front of the art pieces) would be useful to discard profiles which are quite unlikely and may lead to artificial, yet practically hard instances; this generation nevertheless requires to perform a time-consuming data collection and analysis. Another perspective is to relax the full additive independence assumption on the utility function, in order to take account of possible synergetic effects. Generalized additive independence could be targeted, requiring sophisticated and complex optimization techniques to be designed and implemented. Our perspective is to take advantage of collaborative filtering techniques and feedback analyses to improve our preference elicitation procedure and derive utility models that are quite good in practice. Taking into account graph decomposition techniques within PMV is also an important issue to consider. Finally, partnering with a museum is on-going. A prototype, Tech-A-Way, has already been developed.

[3]Some sample benchmarks in OPB and PDDL formats and some sample dot files for the two museums are available at http://www.cril.fr/PMV/.

6 Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments and Vincent Vidal for his advices on state-of-the-art planners. This work has partly been supported by the Conseil Régional du Nord/Pas-de-Calais and an EC FEDER grant.

References

2012. Palais des Beaux Arts de Lille website. http://www.pba-lille.fr/.

2012. IBM CPLEX Optimizer. http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/.

Fox, M., and Long, D. 2003. PDDL2.1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research* 20:61–124.

Gansner, E. R., and North, S. C. 2000. An open graph visualization system and its applications to software engineering. *SOFTWARE - PRACTICE AND EXPERIENCE* 30(11):1203–1233. http://www.graphviz.org/.

Garcia, A.; Arbelaitz, O.; Linaza, M. T.; Vansteenwegen, P.; and Souffriau, W. 2010. Personalized tourist route generation. In *Proceedings of the 10th International Conference on Current Trends in Web Engineering (ICWE'10)*, 486–497.

Gebser, M.; Kaufmann, B.; Neumann, A.; and Schaub, T. 2007. Conflict-driven answer set solving. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, 386–392.

Gerevini, A.; Saetti, A.; and Serina, I. 2003. Planning through stochastic local search and temporal action graphs in LPG. *Journal of Artificial Intelligence Research* 20:239–290.

Hage, W. v.; Stash, N.; Wang, Y.; and Aroyo, L. 2010. Finding your way through the Rijksmuseum with an adaptive mobile museum guide. In *Proceedings of the 7th Extended Semantic Web Conference (ESWC'10)*, 46–59.

Jaén, J.; Mocholì, J. A.; Català, A.; and Navarro, E. 2011. Digital ants as the best cicerones for museum visitors. *Applied Soft Computing* 11(1):111–119.

Kabassi, K. 2010. Personalizing recommendations for tourist. *Telematics and Informatics* 27(1):51–66.

Karp, R. M. 1972. Reducibility Among Combinatorial Problems. In Miller, R. E., and Thatcher, J. W., eds., *Complexity of Computer Computations*. Plenum Press. 85–103.

Le Berre, D., and Parrain, A. 2010. The Sat4j library, release 2.2 system description. *Journal on Satisfiability, Boolean Modeling and Computation* 7:59–64.

Luby, M.; Sinclair, A.; and Zuckerman, D. 1993. Optimal speedup of Las Vegas algorithms. *Information Processing Letters* 47:173–180.

Manquinho, V. M., and Roussel, O. 2006. The first evaluation of pseudo-Boolean solvers (PB'05). *Journal on Satisfiability, Boolean Modeling and Computation* 2(1-4):103–143.

2012. The National Gallery (London) website. http://www.nationalgallery.org.uk/.

Vansteenwegen, P.; Souffriau, W.; Berghe, G. V.; and Oudheusden, D. V. 2011. The city trip planner: An expert system for tourists. *Expert Systems with Applications* 38(6):6540–6546.

Vansteenwegen, P.; Souffriau, W.; and Oudheusden, D. V. 2011. The orienteering problem: A survey. *European Journal of Operational Research* 209(1):1–10.

Wang, Y.; Wang, S.; Stash, N.; Aroyo, L.; and Schreiber, G. 2010. Enhancing content-based recommendation with the task model of classification. In *Proceedings of the 17th International Conference on Knowledge Engineering and Management by the Masses (EKAW'10)*, 431–440.