MATHEMATICS OF OPERATIONS RESEARCH
Vol. 34, No. 3, August 2009, pp. 758–768
ISSN 0364-765X | EISSN 1526-5471 | 09 | 3403 | 0758

informs®

DOI 10.1287/moor.1090.0398
© 2009 INFORMS

Refined MDP-Based Branch-and-Fix Algorithm for the Hamiltonian Cycle Problem

Vladimir Ejov, Jerzy A. Filar, Michael Haythorpe, Giang T. Nguyen
Centre for Industrial and Applied Mathematics, University of South Australia, Mawson Lakes, South Australia 5095, Australia
{vladimir.ejov@unisa.edu.au, http://people.unisa.edu.au/vladimir.ejov;
jerzy.filar@unisa.edu.au, http://people.unisa.edu.au/jerzy.filar;
michael.haythorpe@unisa.edu.au, http://people.unisa.edu.au/michael.haythorpe;
giang.nguyen@unisa.edu.au, http://people.unisa.edu.au/giang.nguyen}

We consider the famous Hamiltonian cycle problem (HCP) embedded in a Markov decision process (MDP). More specifically, we consider the HCP as an optimisation problem over the space of occupation measures induced by the MDP's stationary policies. In recent years, this approach to the HCP has led to a number of alternative formulations and algorithmic approaches. In this paper, we focus on a specific embedding, because of the work of Feinberg. We present a "branch-and-fix" type algorithm that solves the HCP. At each branch of the algorithm, only a linear program needs to be solved and the dimensions of the successive linear programs are shrinking rather than expanding. Because the nodes of the branch-and-fix tree correspond to specially structured 1-randomised policies, we characterise the latter. This characterisation indicates that the total number of such policies is significantly smaller than the subset of all 1-randomised policies. Finally, we present some numerical results.

Key words: Markov decision processes; Hamiltonian cycles; branch-and-bound algorithm
MSC2000 subject classification: Primary: 05C38; secondary: 90C40
OR/MS subject classification: Primary: networks/graphs, branch-and-bound algorithm; secondary: Markov processes
History: Received July 25, 2007; revised September 8, 2008 and March 19, 2009. Published online in *Articles in Advance* August 6, 2009.

1. Preliminaries. This paper is a continuation of a line of research (Filar and Krass [10], Andramonov et al. [2], Filar and Lasserre [11], Feinberg [7]) that aims to exploit the tools of Markov decision processes (MDPs) to study properties of a famous combinatorial optimisation problem, the Hamiltonian cycle problem (HCP). In particular, the nonstandard branch-and-bound method of Filar and Lasserre [11] is applied to Feinberg's [7] embedding of the HCP in a discounted (rather than limiting average used previously) Markov decision process. This is because the discount parameter does not destroy sparsity of the coefficient matrices to nearly the same extent as did the perturbation parameter ε in Filar and Lasserre [11]. In the process, the structure of 1-randomised policies is fully explained. More specifically, the present paper provides evidence that computationally effective algorithms for determining Hamiltonicity can be developed based on this approach. As such, it can also be viewed as a continuation of the numerical experiments begun in Andramanov et al. [2] and continued in Ejov et al. [5], Filar and Lasserre [11], and Eshragh et al. [6]. For a recent survey of this line of research, the reader is referred to Filar [9].

We consider the following version of the HCP: *Given a directed graph, find a simple cycle that contains all vertices of the graph* (*Hamiltonian cycle*) *or prove that a Hamiltonian cycle* (*HC*) *does not exist*. With respect to this property—*Hamiltonicity*—graphs possessing one or more Hamiltonian cycles are called *Hamiltonian*.

Many of the successful classical approaches of discrete optimisation focus on solving a linear programming "relaxation" followed by heuristics that prevent the formation of subcycles. In our approach, we embed a given graph in an MDP in such a way that we can identify HCs and subcycles with exhaustive and nonexhaustive ergodic classes of induced Markov chains.

More precisely, our dynamic, stochastic approach to the HCP considers a moving object tracing out a directed path on the graph G with its movement "controlled" by a function f, called a *policy*. Let $\mathcal{A}$ denote the set of arcs of the graph and $\mathcal{S} = \{1, \ldots, N\}$ be the set of nodes. Thus, a policy f selects an arc $(i, a) \in \mathcal{A}$ at each node $i \in \mathcal{S}$. This selection can be made either in a deterministic or in a randomized way. If it is done in a deterministic way, then $f\colon \mathcal{S} \to \mathcal{S}$ with the convention $f(i) = a$. The set of nodes $\mathcal{S}$ will also serve as the *state space* of an MDP Γ and at each node $i \in \mathcal{S}$, the *action space* $\mathcal{A}(i) := \{a \mid (i, a) \in \mathcal{A}\}$ is in one-to-one correspondence with the set of arcs emanating from that node, or, equivalently, with the set of endpoints of those arcs. From now on, node 1 is called the *home node*.

Illustration. Consider the complete graph G_5 on five nodes (with no self-loops) and think of the nodes as the states of an MDP, denoted by Γ, and of the arcs emanating from a given node as actions available at that state. In a natural way, the standard HC $c_1\colon 1 \to 2 \to 3 \to 4 \to 5 \to 1$ corresponds to the "deterministic policy" $f_1\colon \{1, 2, 3, 4, 5\} \to \{2, 3, 4, 5, 1\}$, where $f_1(2) = 3$ corresponds to the controller choosing the arc $(2, 3)$ in state 2 with probability 1. The Markov chain induced by f_1 is given by the "zero-one" transition matrix $\mathbf{P}(f_1)$, which, clearly, is irreducible. On the other hand, the union of two subcycles: 1 → 2 → 3 → 1 and 4 → 5 → 4 corresponds to the policy f_2: $\{1,2,3,4,5\} \rightarrow \{2,3,1,5,4\}$, which identifies the transition matrix $\mathbf{P}(f_2)$ (see below) containing two distinct ergodic classes:

$$\mathbf{P}(f_2) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

This leads to a natural embedding of the HCP in an MDP Γ. In this topic, the type of MDP that results is identified by an evaluation criterion used to assess a policy's performance. Two of the most commonly used criteria are the so-called *discounted* and *limiting average* evaluation criteria. In this paper, we shall focus on the former, where the performance of any given policy f is calculated by the formula

$$v_\beta(e_i, f) := E_{e_i}^{f} \sum_{t=0}^{\infty} \beta^t r(i_t, a_t), \tag{1}$$

where $e_i = (0, \dots, 1, \dots, 0)$ is the initial distribution vector with 1 in the ith element, (i_t, a_t) is the state-action (node-arc) pair at stage t, $r(i_t, a_t)$ is the reward accrued from such a pair at the tth stage, $\beta \in [0, 1)$ is the *discount factor*, and E denotes the expectation operator.

It will be convenient to work in the so-called *occupation measure space* $\mathcal{X}_\beta$. In that space, we introduce an optimisation problem and develop a "branch-and-fix" type algorithm for solving the HCP. We prove a theoretical property that explains why, in many instances, it is reasonable to expect this algorithm to converge quickly.

We also report on a series of numerical experiments with this algorithm to test its effectiveness. In practice, the algorithm reduces the number of possible branches. We achieve further significant improvements by introducing additional feasibility constraints as bounds and logical checks to limit the number of branches. The number of branches required to find HCs was generally reduced to a tiny fraction of the total number of deterministic policies. For example, a 24-node Hamiltonian cubic graph has $3^{24} \approx 3 \times 10^{11}$ possible choices for deterministic policies, but the algorithm found an HC by examining 12 branches. Hamiltonian graphs performed better than non-Hamiltonian graphs because they generally have many HCs and only one needs to be found. However, even in non-Hamiltonian graphs the algorithm performed rather well. For example, a 28-node non-Hamiltonian cubic graph has $3^{28} \approx 2 \times 10^{13}$ possible choices for deterministic policies, but the algorithm terminated after only 9,631 branches.

2. A formulation of HCP by means of a discounted MDP.

The fact that HCP can be embedded in a natural way in a Markov decision process was demonstrated in Filar and Krass [10], Feinberg [7], and elsewhere. However, in Filar and Krass [10], the long-run average MDP was used, whereas Feinberg [7] exploited the discounted MDP for the first time.

For a given graph G, an *adjacency matrix* $\mathbf{G}$ is defined as

$$[\mathbf{G}]_{ia} = \begin{cases} 1, & \text{if } (i, a) \in \mathcal{A}, \\ 0, & \text{otherwise.} \end{cases}$$

We now formally introduce the transition probabilities for Γ defined by

$$p(j \mid i, a) = \begin{cases} 1, & \text{if } a = j, \\ 0, & \text{otherwise.} \end{cases}$$

Here, $p(j \mid i, a)$ represents the probability of moving from i to j by choosing the action/arc (i, a), so that

$$\sum_{j=1}^{N} p(j \mid i, a) = 1 \quad \text{for all } (i, a) \in \mathcal{A}. \tag{2}$$

A *policy* has entries

$$f_{ia} = \begin{cases} \text{probability of action } a \text{ in state } i, & a \in \mathcal{A}(i), \\ 0, & a \notin \mathcal{A}(i). \end{cases}$$

A policy is *deterministic* if $f_{ia} \in \{0, 1\}$ for all i and a. That is, for each i the controller chooses some particular action $a \in \mathcal{A}(i)$ with probability 1 whenever i is visited. Recall that in this case we also write $f(i) = a$. Let $\mathcal{F}_{\mathcal{D}}$ be the set of all deterministic policies on G. A policy is *stationary* if the controller's decision does not depend on the time of a visit to the state i. Let $\mathcal{F}_{\mathcal{S}}$ be the set of all feasible stationary policies on G. Any stationary policy f gives rise to an $N\times N$ *probability transition matrix* $\mathbf{P}(f)$ with entries

$$[\mathbf{P}(f)]_{ij} := p_{ij}(f) := \sum_{a=1}^{N} p(j\mid i,a)f_{ia},$$

where

$$\sum_{j=1}^{N} p_{ij}(f) = 1 \quad \text{for all } i\in\mathcal{S}.$$

It is well known that the matrix $(\mathbf{I}-\beta\mathbf{P}(f))$ is invertible and that

$$v_\beta(e_i,f) = e_i(\mathbf{I}-\beta\mathbf{P}(f))^{-1}r(f), \tag{3}$$

where $r(f)$ is the immediate expected reward vector induced by the policy f. It is important to note that in the application presented here, for every policy f, we will assume that $r(f)=(1,0,\ldots,0)^T$, an N-dimensional column vector. The latter is used merely to distinguish the home node from any other node.

The *occupation measure space*, $\mathcal{X}_\beta := \{x(f)\mid f\in\mathcal{F}_{\mathcal{S}}\}$ (induced by stationary policies), consists of vectors $x(f)$ whose entries are *occupation measures of the state-action pairs* $(i,a)\in\mathcal{A}(i)$ defined by

$$x_{ia}(f) := \{\nu[(\mathbf{I}-\beta\mathbf{P}(f))^{-1}]\}_i f_{ia}, \tag{4}$$

where $\nu=(\nu_1,\ldots,\nu_N)$ denotes an arbitrary initial state distribution.

In what follows, we shall consider a specially structured initial distribution. Namely, for $\mu\in(0,1/N)$, we define

$$\nu_i = \begin{cases} 1-(N-1)\mu, & \text{if } i=1,\\ \mu, & \text{otherwise.}\end{cases} \tag{5}$$

The *occupation measure of the state* i is defined as the aggregate

$$x_i(f) := \sum_{a\in\mathcal{A}(i)} x_{ia}(f) = \{\nu[\mathbf{I}-\beta\mathbf{P}(f)]^{-1}\}_i, \tag{6}$$

where the second-last equality follows from the fact that $\sum_{a\in\mathcal{A}(i)} f_{ia}=1$. In particular,

$$x_1(f) := \sum_{a\in\mathcal{A}(1)} x_{1a}(f) = \{\nu[\mathbf{I}-\beta\mathbf{P}(f)]^{-1}\}_1 = v_\beta(\nu,f). \tag{7}$$

The construction of x in (4) defines a map M of the policy space $\mathcal{F}_S$ into $\mathbb{R}^{|\mathcal{A}|}$ by

$$M(f) := x(f).$$

It is well known (e.g., see Hordijk and Kallenberg [14] and Filar and Vrieze [12]) that for $\nu>0$, the map M is invertible and its inverse M^{-1} is defined by

$$M^{-1}(x)[i,a] = f_x[i,a] := \frac{x_{ia}}{x_i}.$$

It is also known that, in this case, extreme points of $\mathcal{X}_\beta$ are in one-to-one correspondence with deterministic policies of Γ. However, this important property is lost when entries of ν are permitted to take on zero values. We now recall (Filar and Vrieze [12]) the partition of the space $\mathcal{F}_{\mathcal{D}}$ of deterministic strategies that is based on the subgraphs they trace out in G. In particular, note that with each $f\in\mathcal{F}_{\mathcal{D}}$, we associate a subgraph G_f of G defined by

$$\operatorname{arc}(i,a)\in G_f \iff f(i)=a.$$

We shall also denote a simple cycle of length k and beginning at 1 by a set of arcs

$$c_k^1 = \{(i_1=1,i_2),(i_2,i_3),\ldots,(i_k,i_{k+1}=1)\}, \quad k=2,3,\ldots,N.$$

Thus, c_N^1 is an HC. If G_f contains a cycle c_k^1, we write $G_f\supset c_k^1$. Let

$$C_k^1 := \{f\in\mathcal{F}_{\mathcal{D}}\mid G_f\supset c_k^1\},$$

namely, the set of deterministic policies that trace out a simple cycle of length k, beginning at 1, for all $k=2,3,\ldots,N$. Thus, $\bigcup_{k=2}^{N} C_k^1$ contains all deterministic policies that define a cycle containing node 1. We shall refer to the policies in the union $\bigcup_{k=2}^{N-1} C_k^1$ as *short cycles* (see Figure 1). Denote the complement of $\bigcup_{k=2}^{N} C_k^1$ in $\mathcal{F}_D$ by $\mathcal{N}_c$. Then, $\mathcal{N}_c$ will contain policies that start at the home node 1, and the node where the strategy for the first time repeats itself is different from node 1. Such policies we will call *noose cycles* (see Figure 2). Note that the home node will be a transient state in a Markov chain induced by a noose-cycle policy.

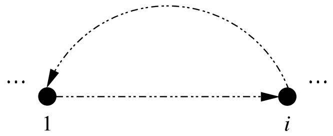

FIGURE 1. A short cycle.

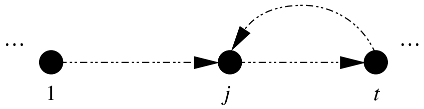

FIGURE 2. A noose cycle.

3. Structure of 1-randomised policies. Much of the analysis in this paper depends on the following proposition, the main part of which was proved in Feinberg [7].

PROPOSITION 3.1. *Let $\beta \in [0,1)$ and $\mu \in [0, 1/N)$. Suppose that $f \in \mathcal{F}_S$ is Hamiltonian. The following two properties hold*:

(i) *If the initial state distribution ν is given by*

$$\nu_i = \begin{cases} 1-(N-1)\mu, & \textit{if } i=1, \\ \mu, & \textit{otherwise}, \end{cases}$$

then

$$v_\beta(\nu, f) = \frac{(1-(N-1)\mu)(1-\beta)+\mu(\beta-\beta^N)}{(1-\beta)(1-\beta^N)}.$$

(ii) *If $f \in \mathcal{F}_{\mathcal{D}}$ and $v_\beta(\nu, f)$ is as in* (i), *then f is a Hamiltonian policy.*

PROOF. For $\mu = 0$, the first part is established in Feinberg [7], and for $\mu \in (0, 1/N)$, this part is merely an extension of the case when $\mu = 0$. The second part also follows by the same argument as the analogous result in Feinberg [7]. □

This proposition, together with the standard conversion of a discounted MDP to a linear program (see Hordijk and Kallenberg [14]), suggests that HCs can be sought among the extreme points of the following subset of the *occupation measure space* $\mathcal{X}_\beta$ that is defined by the linear constraints:

$$\sum_{i=1}^{N} \sum_{a \in \mathcal{A}(i)} (\delta_{ij} - \beta p(j \mid i, a)) x_{ia} = \nu_j, \quad j \in \mathcal{S}, \tag{8}$$

$$\sum_{a \in \mathcal{A}(1)} x_{1a} = \frac{(1-(N-1)\mu)(1-\beta)+\mu(\beta-\beta^N)}{(1-\beta)(1-\beta^N)}, \tag{9}$$

$$x_{ia} \geq 0, \quad i \in \mathcal{S}, \quad a \in \mathcal{A}(i). \tag{10}$$

Let $\overline{\mathcal{X}}_\beta = \{x \mid x \text{ satisfies constraints (8)–(10)}\}$. It is well known (e.g., see Filar and Vrieze [12]) that (when $\mu > 0$) every extreme point of $\mathcal{X}_\beta$ corresponds to a deterministic policy via the transformations M and M^{-1} introduced earlier. Hence, these extreme points must contain exactly N positive entries (one for each node). However, the additional constraint (9) in $\overline{\mathcal{X}}_\beta$ could introduce one more positive entry in its extreme points. This can be seen as follows.

Let x_e be an extreme point of $\overline{\mathcal{X}}_\beta$. It is clear that if x_e contains exactly N positive entries, then by Proposition 3.1, $f = M^{-1}(x_e)$ is a Hamiltonian policy. However, if x_e contains $N+1$ positive entries, then $f = M^{-1}(x_e)$ is a *1-randomised policy* where randomisation occurs only in one state/node, which we call the *splitting node*, and on only two actions/arcs. This terminology was introduced in Feinberg and Shwartz [8].

Clearly, finding an extreme point of $\overline{\mathcal{X}}_\beta$ that corresponds to a deterministic policy solves the HCP. Hence, it is important to understand the structure of these 1-randomised extreme points because they are the ones preventing a trivial LP solution to the HCP. The remainder of this section is devoted to this task.

Let f_α be a 1-randomised policy induced by an extreme point of $\overline{\mathcal{X}}_\beta$, where randomisation only occurs at exactly one node $i \in \mathcal{S}$, over two actions that are chosen with probability α and $1-\alpha$, respectively, for $\alpha \in (0,1)$. Any such 1-randomised policy f can be decomposed into two deterministic policies f_1 and f_2 such that $f = \alpha f_1 + (1-\alpha) f_2$, and the splitting node will be denoted by i.

The next result shows that these 1-randomised policies must always be simple convex combinations of short-cycle and noose-cycle policies. Our proof makes use of Lemma 8.2 in Altman [1], which has recently been generalized by Denardo et al. [4].

THEOREM 3.1. *For some $\mu_0 \in (0, 1/N)$ and for any $\mu \in [0, \mu_0)$, define the initial distribution $\nu = (1-(N-1)\mu, \mu, \ldots, \mu)$. Let an extreme point of $\mathscr{X}_\beta$ induce a 1-randomised policy f_α via the transformation M^{-1}, $\alpha \in (0,1)$, and f_1, f_2 be two deterministic policies that share the same action at all nodes except the splitting node i, such that $f_\alpha = \alpha f_1 + (1-\alpha) f_2$. The following two properties hold:*

(i) *One of the policies $\{f_1, f_2\}$ is a short-cycle policy, and the other one is a noose-cycle policy.*

(ii) *For every such pair $f_1 \in \bigcup_{k=2}^{N-1} C_k^1$ and $f_2 \in \mathscr{N}_c$, there is only one particular value of α such that $x(f_\alpha)$ is an extreme point of $\mathscr{X}_\beta$. When $\mu = 0$, this special value of α is given either by (20) or (23) depending on one of only two cases that may arise.*

PROOF. Let $\mathbf{P}_1$, $\mathbf{P}_2$, and $\mathbf{P}_\alpha$ be the probability transition matrices induced by the deterministic policies f_1, f_2, and the 1-randomised policy f_α, respectively. From (7), it follows that

$$\sum_{a \in \mathscr{A}(1)} x_{1a} = (1-N\mu)[(\mathbf{I}-\beta\mathbf{P}_\alpha)^{-1}]_{11} + \mu \sum_{i=1}^{N} [(\mathbf{I}-\beta\mathbf{P}_\alpha)^{-1}]_{i1} = v_\beta(\nu, f_\alpha). \tag{11}$$

From Lemma 8.2 in Altman [1], it follows that for some $\lambda \in (0,1)$, the following equality holds:

$$v_\beta(\nu, f_\alpha) = \lambda v_\beta(\nu, f_1) + (1-\lambda) v_\beta(\nu, f_2), \tag{12}$$

for an arbitrary state distribution ν and, in particular, for ν prescribed by (5). We are going to prove that for some $\mu_0 \in (0, 1/N)$ and for all $\mu \in [0, \mu_0)$,

$$v_\beta(\nu, f^n) < v_\beta(\nu, f^h) < v_\beta(\nu, f^s), \tag{13}$$

where f^n, f^h, and f^s are noose-cycle, Hamiltonian, and short-cycle policies, respectively.

For $\mu = 0$, we have $\nu = e_1$. Then, (13) follows because $v_\beta(e_1, f^n) = 1, v_\beta(e_1, f^h) = 1/(1-\beta^N)$, and $v_\beta(e_1, f^s) = 1/(1-\beta^k)$ for $k = 2, \ldots, N-1$, where k is the size of the cycle going through 1 in f^s. Now it is sufficient to show that there exists $\mu_0 \in (0, 1/N)$ such that for all $\mu \in (0, \mu_0)$, (13) holds. However, the latter follows immediately from a standard limiting argument and the fact that there are only a finite number of short cycles and noose cycles.

From (13), it is straightforward that for any pair of deterministic policies f_1 and f_2 to satisfy $\lambda v_\beta(\nu, f_1) + (1-\lambda) v_\beta(\nu, f_2) = v_\beta(\nu, f^h)$, both f_1 and f_2 have to be Hamiltonian cycles, or one of the policies $\{f_1, f_2\}$ has to be a short cycle and the other one has to be a noose cycle. Note that a policy $f_\alpha = \alpha f_1^h + (1-\alpha) f_2^h$, where f_1^h and f_2^h are distinct Hamiltonian policies, would contain more than one randomisation and therefore cannot be a 1-randomised policy. Consequently, by (12) and by (13), we obtain part (i).

For part (ii), in this paper, we will only consider the case where $\mu = 0$, and consequently, $\nu = e_1$. For the proof of analogous results for $\mu > 0$, the interested reader is referred to Nguyen [16]. Let k (for $2 \le k < N$) be the size of the cycle passing through state 1 in f_1, which we call the k-cycle, $j \ne 1$ be the first node f_2 revisits, and m be the size of the noose cycle passing through j in f_2, which we call the m-cycle. From Proposition 3.1(i) and (12), the splitting probability

$$\lambda = \frac{v_\beta(e_1, f_\alpha) - v_\beta(e_1, f_2)}{v_\beta(e_1, f_1) - v_\beta(e_1, f_2)} \tag{14}$$

is unique, where $v_\beta(e_1, f_\alpha) = v_\beta(e_1, f^h) = 1/(1-\beta^N)$. Hence, from (14), we have

$$\lambda = \frac{1/(1-\beta^N) - 1}{1/(1-\beta^k) - 1} = \frac{\beta^N(1-\beta^k)}{\beta^k(1-\beta^N)}. \tag{15}$$

In our setting, the derivation of (12) presented in Lemma 8.2 in Altman [1] implies that

$$\lambda = \frac{\alpha(1-\bar{p}(f_1; i, i))}{1-\bar{p}(f_\alpha; i, i)}, \tag{16}$$

where $\bar{p}(f; i, i)$ is the probability of ever returning to i after starting from i, using a policy f. In Altman [1], a transient MDP with a total reward criterion is used that we can induce here by defining $\widehat{\mathbf{P}} := \beta \mathbf{P}(f)$ for any stationary policy f. Hence, for the short-cycle (deterministic) policy f_1, it is clear that

$$\bar{p}(f_1; i, i) = \beta^k. \tag{17}$$

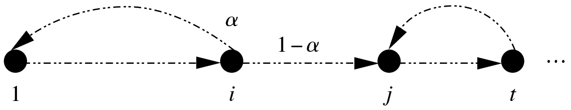

Figure 3. Case 1.

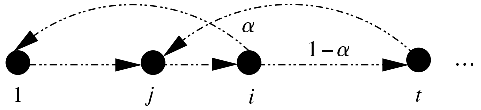

Figure 4. Case 2: $j \neq i$.

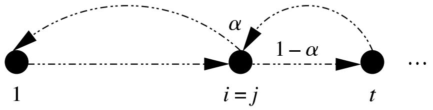

Figure 5. Case 2: $j = i$.

Now, the 1-randomised policy f_α has one of the following structures:
(1) j is not on the k-cycle (see Figure 3);
(2) j is on the k-cycle (see Figures 4 and 5).
Note that because j is the first node f_2 revisits, j will occur infinitely often under f_2. Hence, it is not possible for node j to be on the path from i to 1 because then from j, f_2 would continue to 1, and the home node would be the first node f_2 revisits, which is a contradiction.

Case 1. Suppose that j is not on the k-cycle. Then, using f_α, after starting at the splitting node, a return to i only occurs if we follow the k-cycle. Therefore,

$$\bar{p}(\beta f_\alpha; i, i) = \alpha\beta^k. \tag{18}$$

By (16), (17), and (18), we have

$$\frac{\beta^N(1-\beta^k)}{\beta^k(1-\beta^N)} = \frac{\alpha(1-\beta^k)}{1-\alpha\beta^k}. \tag{19}$$

Therefore,

$$\alpha = \frac{\beta^N}{\beta^k}. \tag{20}$$

Case 2. Suppose that j is on the k-cycle. Using f_α, in order to return to i after leaving i, the system has to follow either the k-cycle or the m-cycle. Hence,

$$\bar{p}(\beta f_\alpha; i, i) = \alpha\beta^k + (1-\alpha)\beta^m. \tag{21}$$

By (16), (17), and (21), we have

$$\frac{\beta^N(1-\beta^k)}{\beta^k(1-\beta^N)} = \frac{\alpha(1-\beta^k)}{1-\alpha\beta^k-(1-\alpha)\beta^m}. \tag{22}$$

Therefore,

$$\alpha = \frac{\beta^N-\beta^{N+m}}{\beta^k-\beta^{N+m}}. \quad \square \tag{23}$$

4. Branch-and-fix method. In view of the fact that it is “only” the 1-randomised policies that prevent us from using an LP-method to find an HC, it has been recognised for some time that branch-and-bound type methods could be used to eliminate the possibility of arriving at these undesirable extreme points (e.g., see Filar and Lasserre [11]). However, the method reported in Filar and Lasserre [11] used an embedding in a long-run average MDP with a perturbation of transition probabilities that introduces a small parameter in some coefficients of variables in linear constraints (8). Furthermore, that method was never implemented or tested beyond a few simple examples.

Theorem 3.1 from the preceding section now indicates that 1-randomised policies induced by extreme points of $\overline{\mathscr{X}}_\beta$ are less prevalent than might have been conjectured because they cannot be made of convex combinations of just any two deterministic policies. This provides motivation for testing algorithmic approaches based on successive elimination of arcs that could be used as "bridges" to construct these convex combinations.

Note that because our ultimate goal is to find an extreme point $x_e \in \overline{\mathscr{X}}_\beta$ such that

$$f = M^{-1}(x_e) \in \mathscr{F}_D,$$

we have a number of degrees of freedom in designing an algorithm. In particular, different linear objective functions can be chosen at each stage of the algorithm, the parameter $\beta \in (0,1)$ can be adjusted, and $\mu \in (0, 1/N)$ can be chosen small, but not so small as to cause numerical difficulties. This parameter needs to be positive to ensure that M^{-1} is well defined. In the experiments reported here, $\mu = 1/N^2$.

The logical structure of our branch-and-fix[1] (B&F) algorithm is as follows. We will be solving a sequence of linear programs—two at each node of the B&F "tree"—with the generic structure:

$$\begin{aligned} \min\quad & L(x) \\ \text{subject to:}\quad & x \in \overline{\mathscr{X}}_\beta, \\ & \text{additional constraints, if any, on arcs fixed earlier.} \end{aligned} \tag{$\dagger$}$$

Step 1 (*Initiation*). Solve the original LP (†) without any additional constraints and with some choice of an objective function $L(x)$. Let x_0 be an optimal basic feasible solution obtained. Find $f_0 := M^{-1}(x_0)$. If $f_0 \in \mathscr{F}_D$, stop; the policy f_0 identifies an HC.

Step 2 (*Branching*). Use the 1-randomised policy f_0 to identify a node i and two arcs (i, j_1) and (i, j_2) corresponding to the single randomisation in f_0. If there are d arcs $\{(i, a_1), \ldots, (i, a_d)\}$ emanating from node i, construct subgraphs: $G_1, G_2, \ldots, G_d$, where in G_k the arc (i, a_k) is the only arc emanating from node i. These graphs are identical to the original graph G at all other nodes. Note that this process already "fixes" an arc in each G_k.

Step 3 (*Fixing*). It turns out that in many graphs fixing one arc implies that other arcs can also be fixed.[2] A number of checks for determining additional arcs that can be fixed are described later in this section. These are the arcs that are fixed at this step in the current implementation of the algorithm.

Step 4 (*Iteration*). Repeat Step 1 with the LP (†) constructed for the graph at the current node. Note that this node may correspond to $G_1, G_2, \ldots, G_d$, or to a subgraph constructed from one of these with the help of additional arc fixing.[3]

If f is a Hamiltonian policy, $x = M(f)$, and $\mu = 0$, then it can be easily checked that

$$x_{i_k i_{k+1}} = \sum_a x_{i_k a} = \frac{\beta^{k-1}}{1-\beta^N},$$

where (i_k, i_{k+1}) is the kth arc on the HC traced out by f. This immediately suggests lower and upper bounds on sums of the x-variables corresponding to arcs emanating from the "heads" of fixed arcs. This is because it is clear that if $i_{k+1} \neq 1$,

$$\sum_a x_{i_{k+1} a} - \beta x_{i_k i_{k+1}} = 0.$$

For $\mu > 0$, analogous (but somewhat more complex) expressions for the preceding sums can be derived, and the above relationship between these sums at successive nodes on the HC is simply

$$\sum_a x_{i_{k+1} a} - \beta x_{i_k i_{k+1}} = \mu. \tag{24}$$

For the final arc $(i_N, 1)$, we have

$$-\beta^N \sum_a x_{1a} + \beta x_{i_N 1} = \frac{\mu\beta(1-\beta^{N-1})}{1-\beta}. \tag{25}$$

[1] Because the speed of convergence depends more on the arc-fixing features than on the bounds, the name "branch-and-fix" (or B&F) algorithm is more appropriate than "branch-and-bound."

[2] For example, this frequently happens in the case of cubic graphs that supplied many of our test examples.

[3] As is typical with branching methods, decisions guiding which branch to select first are important and open to alternative heuristics.

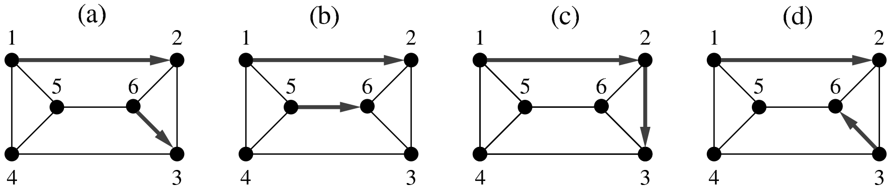

FIGURE 6. Various arc fixing situations.

Structure of LP (†) in one implementation. At the initiation step, we solve a feasibility problem of satisfying constraints (8), (9), and (10). This allows us to determine from which node to begin branching.

In every iteration other than the first, we solve an additional LP to determine whether we need to continue exploring the current branch. As the algorithm evolves along successive nodes of the B&F tree, we have additional information about which arcs had been fixed, permitting us to perform tests as to whether there remains a possibility of finding an HC that incorporates these arcs. If that is impossible, that node of the B&F tree can be fathomed and no further exploration of that branch is needed.

For example, suppose that fixed arcs (i, j) belong to a set $\mathcal{U}$. Let the objective function of a second LP (‡) be $L(x) = \sum_a x_{1a}$ and minimise $L(x)$ subject to constraints (8) and (10), together with Equations (24) and (25), providing additional constraints for each arc in $\mathcal{U}$. If the minimum fails to reach the level set by the right-hand side of the now-omitted constraint (9) of $\mathcal{X}_\beta$, then no HC exists that uses all the arcs of $\mathcal{U}$. Otherwise, we solve the LP (†) with the objective function[4]

$$L(x) = \sum_{(i,j)\in\mathcal{U}} \left[\sum_{a\in\mathcal{A}(j)} x_{ja} - \beta \sum_{a\in\mathcal{A}(i)} x_{ia} \right],$$

and with no additional constraints beyond those in $\mathcal{X}_\beta$. This last LP will either enable us to fathom the current node of the B&F tree, or it will lead to an extreme point x'_e such that $f' = M^{-1}(x'_e)$ is a new 1-randomised policy.

Arc-fixing checks. We found that in many test problems there are a number of logical checks that enable us to fix additional arcs once a decision is made to fix one particular arc. This is a consequence of the fact that when there are nodes with low in/out-degrees, fixing one arc may have immediate implications on what arcs must go into (or out of some other nodes). This is best illustrated with the help of an example. Note that these checks are in the spirit of well-known rules for constructing HCs (see, for example, §8.2 of Tucker [17]).

Consider the simple 6-node cubic "envelope" graph. Figure 6 shows the kind of logical additional arc fixing that can arise.

Check 1. Consider Figure 6(a). So far, the only fixed arcs are $(1, 2)$ and $(6, 3)$. Because the only arcs that can go to node 5 are $(1, 5)$, $(4, 5)$, and $(6, 5)$, we must fix arc $(4, 5)$ because nodes 1 and 6 already have fixed arcs going elsewhere. In general, if there is only one arc available to go to a node, it must be fixed.

Check 2. Consider Figure 6(b). So far, the only fixed arcs are $(1, 2)$ and $(5, 6)$. The only arcs left going to node 5 are $(4, 5)$ and $(6, 5)$, but we cannot choose $(6, 5)$ because this will create a subcycle, so we must fix arc $(4, 5)$. In general, if there are only two arcs available to go to a node, and one will create a loop, we must fix the other one.

Check 3. Consider Figure 6(c). So far, the only fixed arcs are $(1, 2)$ and $(2, 3)$. Because the only arcs that can come from node 6 are $(6, 2)$, $(6, 3)$, and $(6, 5)$, we must fix arc $(6, 5)$ because nodes 2 and 3 already have arcs going into them. In general, there is only one arc available to come from a node, so it must be fixed.

Check 4. Consider Figure 6(d). So far, the only fixed arcs are $(1, 2)$ and $(3, 6)$. The only arcs that can come from node 6 are $(6, 2)$, $(6, 3)$, and $(6, 5)$, but $(6, 2)$ will repeat a node and we cannot choose $(6, 3)$ because this will create a subcycle, so we must fix arc $(6, 5)$. In general, if there are only three arcs available to come from a node, and one will repeat a node whereas another will create a loop, we must fix the third one.

5. Numerical results.

The above algorithm was implemented in MATLAB (version 7.4.0.336), and used CPLEX (version 11.0.0) to solve all the linear programming subproblems. The algorithm was tested on a range of relatively small graphs. The results were encouraging. The number of branches required to solve each of

[4] For simplicity, we are assuming here that $\mathcal{U}$ does not contain an arc going into node 1. If such an arc were in $\mathcal{U}$, this objective would have one term consistent with the left-hand side of Equation (25).

Table 1. Preliminary results for the refined branch-and-fix method.

Graph	Branches	Upper bound	Time
Dodecahedron: Hamiltonian $N = 20$	43	3.4868×10^{9}	0:01
8×8 Knight's tour: Hamiltonian $N = 64$	1,355	9.1654×10^{43}	0:15
Petersen: Non-Hamiltonian $N = 10$	63	5.9049×10^{4}	0:01
Coxeter: Non-Hamiltonian $N = 28$	9,631	2.2877×10^{13}	0:52

Table 2. Performance of the branch-and-fix method over cubic graphs.

Type of graphs	Average branches (min–max)
Hamiltonian $N = 10$	6.647 (4–10)
Hamiltonian $N = 12$	9.713 (2–21)
Non-Hamiltonian $N = 10$	39 (15–63)
Non-Hamiltonian $N = 12$	32.4 (15–102)

these problems was only a tiny fraction of the maximum possible number of branches. It is clear that non-Hamiltonian graphs tend to require more branches to solve than do Hamiltonian graphs of the same size. This can be explained by the fact that a Hamiltonian graph will end branching as soon as an HC is found. The latter does not happen in non-Hamiltonian graphs.

A sample of results can be seen in Tables 1 and 2, including a comparison between the number of branches required and the maximum possible number of branches (upper bound) and the running time in seconds. Note that Dodecahedron, Petersen, Knight's Tour, and Coxeter graphs are well known in the literature (see Gross and Yellen [13, p. 12] for the first two, p. 225 for the third, and Bondy and Murty [3, p. 241] for the last). In the first column of Table 2, we are referring to all connected cubic graphs with the prescribed number of nodes. Hence, in the second column of Table 2, we report only the average number of branches used by the B&F algorithm with the average taken over all graphs in the corresponding class. See Meringer [15] for a reference on generating cubic graphs.

Example 5.1. We conclude the paper with a detailed solution of the simple 6-node envelope graph mentioned earlier, which was solved using the above implementation, with $\beta = 0.99$, $\mu = 1/36$. The example was run using the linear programming solver CPLEX 11.0.0.

We started by solving the following feasibility problem:

$$\begin{aligned}
x_{12} + x_{14} + x_{15} - \beta x_{21} - \beta x_{41} - \beta x_{51} &= 1 - 5\mu,\\
x_{21} + x_{23} + x_{26} - \beta x_{12} - \beta x_{32} - \beta x_{62} &= \mu,\\
x_{32} + x_{34} + x_{36} - \beta x_{23} - \beta x_{43} - \beta x_{63} &= \mu,\\
x_{41} + x_{43} + x_{45} - \beta x_{14} - \beta x_{34} - \beta x_{54} &= \mu,\\
x_{51} + x_{54} + x_{56} - \beta x_{15} - \beta x_{45} - \beta x_{65} &= \mu,\\
x_{62} + x_{63} + x_{65} - \beta x_{26} - \beta x_{36} - \beta x_{56} &= \mu,\\
x_{12} + x_{14} + x_{15} = \frac{\mu\beta(1-\beta^5)}{(1-\beta^6)(1-\beta)} &+ \frac{1-5\mu}{1-\beta^6},\\
x_{ia} \geq 0 \quad \text{for all } i = 1, \ldots, 6, &\quad a \in \mathcal{A}(i).
\end{aligned}$$

The first iteration produced a randomised policy where the randomisation occurred at node 4. The B&F tree then split into three choices: to fix arc $(4, 1)$, $(4, 3)$, or $(4, 5)$. The algorithm first branched on fixing arc $(4, 1)$.

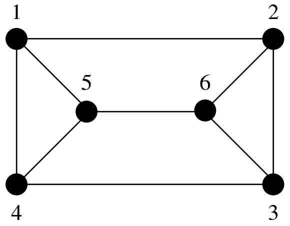

Figure 7. The envelope graph.

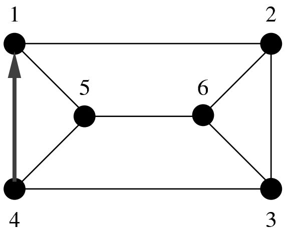

Figure 8. Branching on arc (4, 1).

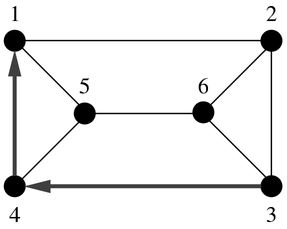

Figure 9. Second branching on arc (3, 1).

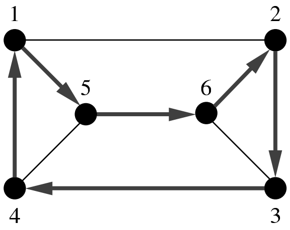

Figure 10. Hamiltonian cycle found by the B&F algorithm.

Because the algorithm uses a depth-first search, the arcs $(4,3)$ and $(4,5)$ would not be fixed unless the algorithm fathomed the $(4,1)$ branch without having found an HC. Note that fixing the arc $(4,1)$ involves eliminating arcs $(4,3)$ and $(4,5)$. In addition, arcs $(1,4)$, $(2,1)$, and $(5,1)$ can also be eliminated because they cannot coexist with arc $(4,1)$ in an HC.

Hence, at the second iteration we solved the following two LPs, the first to check the feasibility of the current fixing of arcs (‡):

$$\begin{aligned}
\min\quad & x_{12}+x_{15}\\
\text{s.t.}\quad & x_{12}+x_{15}-\beta x_{41}=1-5\mu,\\
& x_{23}+x_{26}-\beta x_{12}-\beta x_{32}-\beta x_{62}=\mu,\\
& x_{32}+x_{34}+x_{36}-\beta x_{23}-\beta x_{63}=\mu,\\
& x_{41}-\beta x_{34}-\beta x_{54}=\mu,\\
& x_{54}+x_{56}-\beta x_{15}-\beta x_{65}=\mu,\\
& x_{62}+x_{63}+x_{65}-\beta x_{26}-\beta x_{36}-\beta x_{56}=\mu,\\
& -\beta^6 x_{12}-\beta^6 x_{15}+\beta x_{41}=\frac{\mu(\beta-\beta^6)}{1-\beta},\\
& x_{ia}\ge 0 \quad \text{for all } i=1,\ldots,6,\ \ a\in\mathcal{A}(i).
\end{aligned}$$

Note that the last equality constraint above came from (25) because the fixed arc $(4,1)$ returns to the home node. Because the optimal objective function was equal to the right-hand side of the omitted constraint (9), we could not fathom this branch.

We also solved the updated LP (†):

$$\begin{aligned}
\min\quad & -\beta^6 x_{12}-\beta^6 x_{15}+\beta x_{41}\\
\text{s.t.}\quad & x_{12}+x_{15}-\beta x_{41}=1-5\mu,\\
& x_{23}+x_{26}-\beta x_{12}-\beta x_{32}-\beta x_{62}=\mu,
\end{aligned}$$

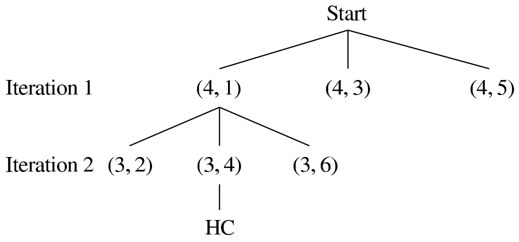

Figure 11. The B&F tree for the envelope graph.

$$
\begin{aligned}
&x_{32}+x_{34}+x_{36}-\beta x_{23}-\beta x_{63}=\mu,\\
&x_{41}-\beta x_{34}-\beta x_{54}=\mu,\\
&x_{54}+x_{56}-\beta x_{15}-\beta x_{65}=\mu,\\
&x_{62}+x_{63}+x_{65}-\beta x_{26}-\beta x_{36}-\beta x_{56}=\mu,\\
&x_{12}+x_{15}=\frac{\mu\beta(1-\beta^5)}{(1-\beta^6)(1-\beta)}+\frac{1-5\mu}{1-\beta^6},\\
&x_{ia}\geq 0 \quad \text{for all } i=1,\dots,6, \quad a\in\mathcal{A}(i).
\end{aligned}
$$

The second iteration produced a randomised policy where the randomisation occurred at node 3. The B&F tree then split into three choices: to fix arc $(3,2)$, $(3,4)$, or $(3,6)$. The algorithm first selected arc $(3,4)$ to branch on.

Examining remaining nodes with multiple (nonfixed) arcs and exploiting Checks 1–4, we immediately see that arc $(5,6)$ must be fixed by Check 3. Repeating the preceding at node 2, we now see that arc $(2,3)$ must be fixed, also by Check 3. Then, at node 6, arc $(6,2)$ must be fixed by Check 4 and, finally, arc $(1,5)$ must be fixed by Check 4. Because this yields the HC $1\to 5\to 6\to 2\to 3\to 4\to 1$, the algorithm terminates.

The whole B&F tree is illustrated in Figure 11.

Acknowledgments. The authors thank W. Murray for interesting discussions and A. Eshragh for detailed reading of the manuscript. An anonymous referee suggested a number of improvements, including the use of a result in Altman [1] and Denardo et al. [4] for the proof of Theorem 3.1. This work was supported by the Australian Research Council Discovery Grant DP0666632.

References

[1] Altman, E. 1999. *Constrained Markov Decision Processes*. Chapman and Hall/CRC, Boca Raton, FL.
[2] Andramonov, M., J. A. Filar, A. Rubinov, P. Pardalos. 2000. Hamiltonian cycle problem via Markov chains and min-type approaches. P. Pardalos, ed. *Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems*. Kluwer Academic Publishers, Dordrecht, The Netherlands.
[3] Bondy, J. A., U. S. R. Murty. 1976. *Graph Theory with Applications*. North-Holland, New York.
[4] Denardo, E. V., E. A. Feinberg, U. G. Rothblum. 2008. On occupation measures for total-reward MDPs. *47th IEEE Conf. Decision and Control*, IEEE, Cancun, Mexico, 4460–4465.
[5] Ejov, V., J. A. Filar, J. Gondzio. 2004. An interior point heuristic for the Hamiltonian cycle problem via Markov decision processes. *J. Global Optim.* **29**(3) 315–334.
[6] Eshragh, A., J. A. Filar, M. Haythorpe. 2009. *A Hybrid Simulation-Optimization Algorithm for the Hamiltonian Cycle Problem*, Springer, Dordrecht, The Netherlands.
[7] Feinberg, E. 2000. Constrained discounted Markov decision process and Hamiltonian cycles. *Math. Oper. Res.* **25**(1) 130–140.
[8] Feinberg, E., A. Shwartz. 1996. Constrained discounted dynamic programming. *Math. Oper. Res.* **21** 922–945.
[9] Filar, J. A. 2007. Controlled Markov chains, graphs, and Hamiltonicity. *Foundations Trends Stochastic Systems* **1**(2) 77–162.
[10] Filar, J. A., D. Krass. 1994. Hamiltonian cycles and Markov chains. *Math. Oper. Res.* **19** 223–237.
[11] Filar, J. A., J.-B. Lasserre. 2000. A non-standard branch and bound method for the Hamiltonian cycle problem. *ANZIAM J.* **42**(E) 556–577.
[12] Filar, J. A., K. Vrieze. 1996. *Competitive Markov Decision Processes*. Springer-Verlag, New York.
[13] Gross, J., J. Yellen. 2005. *Graph Theory and Its Application*. CRC Press, Boca Raton, FL.
[14] Hordijk, A., L. C. M. Kallenberg. 1984. Constrained undiscounted stochastic dynamic programming. *Math. Oper. Res.* **9**(2) 276–289.
[15] Meringer, M. 1999. Fast generation of regular graphs and construction of cages. *J. Graph Theory* **30** 137–146.
[16] Nguyen, G. T. 2009. Hamiltonian cycle problem, Markov decision processes and graph spectra. Ph.D. thesis, University of South Australia, Adelaide.
[17] Tucker, A. 1980. *Applied Combinatorics*. John Wiley & Sons, New York.