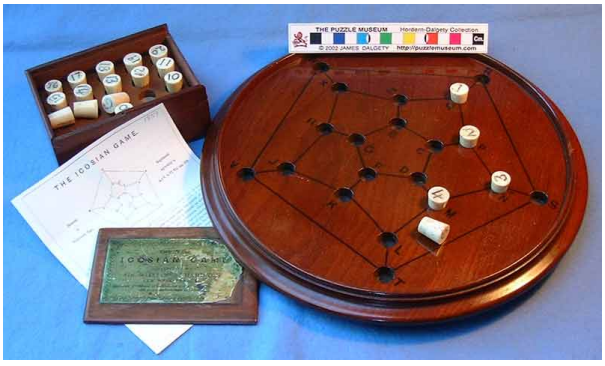
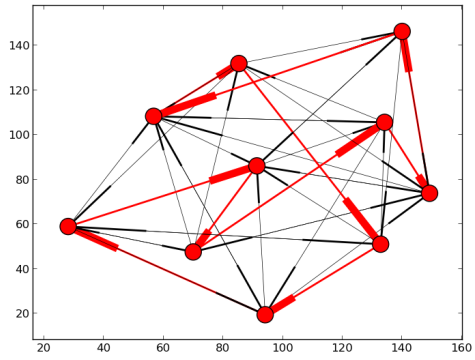
MS&E 315 and CME 304
Hamiltonian cycle problem

1 Overview

The Hamiltonian cycle problem (HCP) is an important graph theory problem that has been studied by mathematicians for many years due to its NP-completeness. It has also gained recognition due to its close relation with famous mathematical problems and puzzles such as the Traveling Salesman Problem (TSP) and the Icosian game. It consists of the following: given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges, determine whether any simple cycle of length $N = |\mathcal{V}|$ exists. A simple cycle, or circuit, is a closed path with no repetition of nodes and edges. Simple cycles of length N are known as *Hamiltonian cycles*. In this project, you will implement an optimization algorithm for finding a Hamiltonian cycle of a graph.



(a) Sir William Rowan Hamilton's Icosian game.



(b) Hamiltonian cycle of a directed graph (in red).

2 Problem formulation

Finding a Hamiltonian cycle of a **directed** graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ can be posed as the optimization problem

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \varphi(x) \\ \text{subject to} & x \in \mathcal{S}. \end{array} \tag{1}$$

The vector x is composed of edge transition probabilities x_{ij}, $(i,j) \in \mathcal{E}$. The objective function φ is given by

$$\varphi(x) = -\det\left(I - P(x) + \frac{1}{N} ee^T\right), \tag{2}$$

where e is the vector of all ones and $P(x)$ is the transition probability matrix

$$P(x)_{ij} = \begin{cases} x_{ij}, & \text{if } (i,j) \in \mathcal{E} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The constraint set $\mathcal{S}$ represents *stochastic* constraints, which restrict the transition probabilities x_{ij} to be non-negative, and the rows of $P(x)$ to sum to one. It can be shown that if x^* is the global optimum of (1), $\varphi(x^*) = -N$ implies that x^* corresponds to a Hamiltonian cycle. Conversely, if $\varphi(x^*) > -N$, the graph does not have a Hamiltonian cycle [2].

Since x^* such that $\varphi(x^*) = -N$ corresponds to a Hamiltonian cycle, the columns of $P(x^*)$ also sum to one. Hence, the optimization can be done over the more restrictive set $\mathcal{DS}$ of *doubly-stochastic* constraints, which are equal to the stochastic constraints with the additional restriction that the columns of $P(x)$ also sum to one. In can be shown that when x remains in $\mathcal{DS}$ and its components removed from zero, only the leading principal minor of $I - P$ is needed to compute the objective function. More specifically,

$$\varphi(x) = -N \det(G^{NN}(x)), \tag{4}$$

where $G(x) = I - P(x)$ and G^{NN} is G with row N and column N removed.

3 Algorithm implementation and analysis

For this project, you have to implement an active set algorithm for solving the HCP problem. It is your choice whether to use second derivatives or not, but we recommend using them and in particular taking advantage of directions of negative curvature. You must try solving the HCP problem in three different ways:

1. Using the doubly stochastic constraints $\mathcal{DS}$.
2. Using only the stochastic constraints $\mathcal{S}$.
3. Using the stochastic constraints $\mathcal{S}$ combined with other constraints that you come up with to try to get better results.

To test the performance of your algorithm with the different problem formulations, you should construct various graphs and determine the fraction of them that are solved with each of the formulations. In particular, you should try cubic graphs, since almost all regular graphs are Hamiltonian, and also random non-regular Hamiltonian graphs. Adequate graph sizes for testing are between 10 to 20 nodes.

4 Report

You are required to write a report in LaTeX describing what you did and the results you obtained. In particular, your report **must** include the following:

1. A description of your optimization algorithm.
2. A justification of why you are confident that your algorithm works correctly.
3. An explanation of the constraints you added to the stochastic constraints to get better results.

4. The results you obtained with each of the different problem formulations.

5. A discussion about the possible reasons for any variation in performance.

You are not required to submit code and may use any programming language you wish.

5 Hints

The following are some hints that may help you do the project.

- You should check the rank of the equality constraints and take any necessary actions if they are rank deficient.
- You should pass your solver an initial feasible point. Such a point should be simple to construct for regular graphs but it requires a bit more work for non-regular ones (you may use routines not written by you for this).
- You should consider rounding the variables to determine if a Hamiltonian cycle has been found before your algorithm has converged.
- You should use finite differences to verify that you have coded the gradient and Hessian of the objective function correctly before trying your solver on the HCP problem.
- You should make your solver show informative output during every iteration.
- You should design your solver as a collection of small and simple submodules and test them separately.

References

[1] W.K. Chen. *Applied Graph Theory*. North-Holland series in applied mathematics and mechanics. Elsevier Science, 2012.

[2] Michael Haythorpe. *Markov Chain Based Algorithms for the Hamiltonian Cycle Problem*. PhD thesis, University of South Australia, July 2010. http://www.stanford.edu/group/SOL/dissertations/michael-haythorpe-thesis.pdf.

[3] Markus Meringer. Fast generation of regular graphs and construction of cages. *J. Graph Theory*, 30(2):137–146, February 1999.

[4] Markus Meringer. Regular graphs, June 2009. http://www.mathe2.uni-bayreuth.de/markus/reggraphs.html.

[5] R. W. Robinson and N. C. Wormald. Almost all regular graphs are Hamiltonian. *Random Struct. Algorithms*, 5(2):363–374, April 1994.

A Objective function and its derivatives

The gradient of the function (2) is given by

$$\nabla\varphi(x)_{(i,j)} = \frac{\partial\varphi(x)}{\partial x_{ij}} = (-1)^{i+j}\det\big(A^{ij}(x)\big), \tag{5}$$

for all $(i,j) \in \mathcal{E}$, where

$$A(x) = I - P(x) + \frac{1}{N}ee^T, \tag{6}$$

and A^{ij} is A with row i and column j removed. The Hessian is given by

$$\nabla^2\varphi(x)_{(i,j),(k,l)} = \frac{\partial^2\varphi(x)}{\partial x_{ij}\partial x_{kl}} = (-1)^{(i+j+\hat{k}+\hat{l}+1)}\det\big(A^{[ij],[kl]}(x)\big), \tag{7}$$

for all (i,j) and $(k,l) \in \mathcal{E}$ such that $i \neq k$ and $j \neq l$, where $A^{[ij],[kl]}$ is A with rows i and k removed, and with columns j and l removed. The quantities $\hat{k}$ and $\hat{l}$ are defined by

$$\hat{k} = \begin{cases} k-1, & \text{if } k > i \\ k, & \text{otherwise,} \end{cases} \tag{8}$$

$$\hat{l} = \begin{cases} l-1, & \text{if } l > j \\ l, & \text{otherwise.} \end{cases} \tag{9}$$

You will find that evaluating (2) and its derivatives using the above formulas is very slow. With function (4), more efficient formulas can be used. In particular, it can be shown that for $x \in \mathbf{relint}(\mathcal{DS})$ (relative interior), an LU factorization $G = LU$ exists without requiring prior permutations and hence that

$$\varphi(x) = -N\prod_{i=1}^{N-1} u_{ii}, \tag{10}$$

where u_{ii} is the i-th diagonal of U. The gradient can be computed with the formula

$$\nabla\varphi(x)_{(i,j)} = -\varphi(x)a_j^T b_i, \tag{11}$$

for all $(i,j) \in \mathcal{E}$, where a_j and b_i are defined by

$$\hat{L}b_i = e_i,\ i = 1,\ldots,N \tag{12}$$

$$\hat{U}^T a_j = e_j,\ j = 1,\ldots,N, \tag{13}$$

$\hat{U}$ is U with the last column replaced by e_N, and $\hat{L}$ is L with the last row replaced by e_N^T. Similarly, the Hessian can be computed efficiently with the formula

$$\nabla^2\varphi(x)_{(i,j),(k,l)} = \eta a_l^T b_i - \nabla\varphi(x)_{(i,j)}a_l^T b_k, \tag{14}$$

for all (i,j) and $(k,l) \in \mathcal{E}$ such that $i \neq k$ and $j \neq l$, where η is defined by

$$\eta = \begin{cases} -\varphi(x)a_j^T b_k, & \text{if } \nabla\varphi(x)_{(k,l)} \neq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$