Winter March 13, 2022

Hamiltonian Circuits as Coalitions and their Core Status

Lester G Telser

2/9/22

Hamiltonian Circuits as Coalitions and their Core Status

L . G . Telser

Definition. Arrows in a Hamiltonian circuit appear only once and have the same orientation.

Let c denote a coalition with m members. Let V[c] denote the value of c. Let x[i] denote the contribution of coalition member i to c.

(1) V[c] = F[X],

X = {x[1], x[2], ... , x[m]}

is an m-vector in the positive orthant of R^n, n > m.

The value of a Hamiltonian circuit in the Nöther algebra is the product of its members' contributions to the circuit.

(2) $F[X] = \prod_{i=1}^{m} x[i]$

Let μ denote the arithmetic mean of X and let γ denote the geometric mean of X.

(3) $\mu = (1/n) \sum_{i=1}^{m} x[i]$ and $\gamma = \sqrt[m]{F[x]}$.

The well-known theorem relating the arithmetic mean to the geometric mean says

(4) $\mu > \gamma$.

The two means are equal if and only if x[i] is the same for all i.

Classical core theory states the following problem.

Let n individuals form coalitions. The minimax theorem applies to the function that defines the value of a coalition. The value is the most the coalition can get under the most adverse conditions. Let N denote the grand coalition of all n individuals. The upper bound on the return to N is V(N). An imputation of the payoffs, r[i], i = 1, 2, ..., n, is in the core if $\sum_{i=1}^{n}$r[i] ≤ V[N], and each coalition c can pay its members at least V[c]. A solution of the following linear programming problem can show whether this is possible (Bondereva [1963]).

Primal Problem Min $\sum_{i=1}^{n} r[i]$ with respect to r[i] ≥ 0 subject to

(5) $\sum_{i \in c} r[i] \geq V[c]$ for all c ∈ S, where S denotes the set of all the

feasible coalitions.
If the constraints have a solution that can satisfy
(6) $\sum_{i \in N} r[i] \leq$ v[N],
then it is in the core. Otherwise, the core is empty.

The problem is whether there is a core for coalitions as Hamiltonian circuits.

Let p[i] = log x[i] denote the payoff to member i in circuit c. Assume the circuit has m members. Equations (2) and (3) imply

(7) $\log \gamma = (1/m) \log [\ \prod_{i=1}^{m} x[i]\ ] = (1/m) \sum_{i=1}^{m} \log x[i]$

$= (1/m) \sum_{i=1}^{m} p[i] = \text{Mean}[p(\text{m})]$.

Inequality (4) implies taking logs gives

(8) $\log \mu > \log \gamma = \text{Mean}\,[\text{p(m)}]$.

The log of the value of circuit c satisfies the following conditions.

(9) $\log \text{V[c]} = \log (\prod_{i=1}^{m} x[i]) = \sum_{i=1}^{m} \log x[i]$

$= \sum_{i=1}^{m} p[i] = \text{m} \log \gamma$.

Therefore,

(10) $\log \text{V[c]} = \text{m} \log \gamma = \text{m Mean}[p(m)]$

These results imply the value of a Hamiltonian circuit is maximal when its members are equally productive.

Equation (10) says; If circuit members are paid the log of their contributions to the circuit, then their payoffs sum to the log of the value of the circuit. If instead they are paid their actual contributions, x[i], then the sum of these payments would exceed the value of the circuit. Also, the sum of these payments over any partition of the individuals into circuits would not be feasible so the sum cannot belong to the core. However, the *log* of the value of the circuit equals the sum of the logs of the member contributions. This satisfies a necessary but not a sufficient condition for a non empty core. One must also show the sum of the logs is maximal so it is an imputation. The maximal valuation occurs only if all members get the same payoff. It is not hard to show the maximal valuation is dominated by the 3-circuit containing the most productive members of the grand coalition.

The best case for a non empty core is when the contribution to a circuit by a member does not depend on the circuit. When it does depend, a non empty core is even less likely.

Some Hamiltonian Circuit Properties

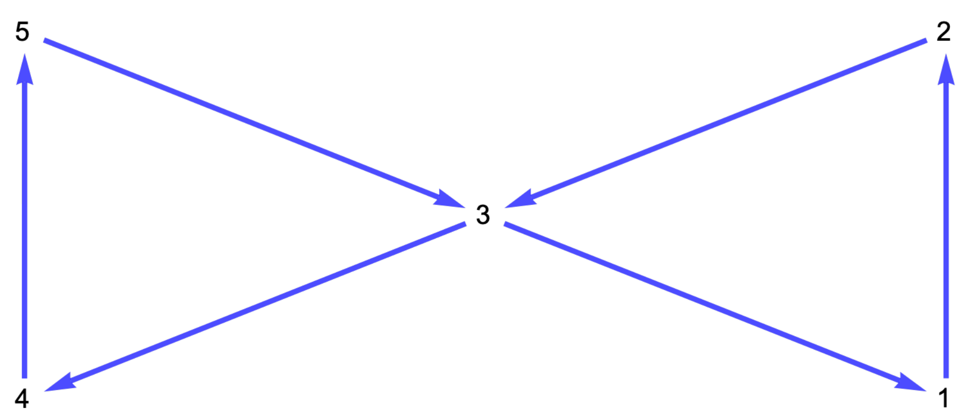
All arrows in a Hamiltonian circuit have the same orientation.
No arrow appears more than once.
A Hamiltonian circuit has no Hamiltonian sub-circuit.
The union of two Hamiltonian circuits is a Hamiltonian circuit if and only if they do not overlap and share only one vertex.
Hamiltonian circuits that satisfy these conditions for their union is an Eulerian Cycle.

In[∘]:=
```
cm := {{0, 1, 0, 0, 0}, {0, 0, 1, 0, 0}, {1, 0, 0, 1, 0},
  {0, 0, 0, 0, 1}, {0, 0, 1, 0, 0}}
```

In[∘]:=
```
gcm := makeGrph[cm, 2]
```

In[∘]:=
```
gcm
```

Out[∘]=



In[∘]:=
```
findSomeCycles[cm, 3]
```

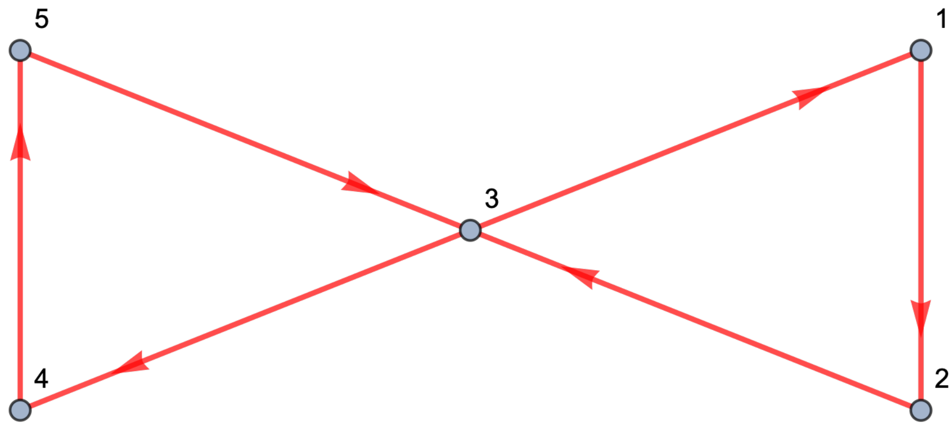
Out[∘]=
```
{{3 ↔ 4, 4 ↔ 5, 5 ↔ 3}, {1 ↔ 2, 2 ↔ 3, 3 ↔ 1}}
```

In[∘]:=
```
findSomeCycles[cm, 6]
```

Out[∘]=
```
{}
```

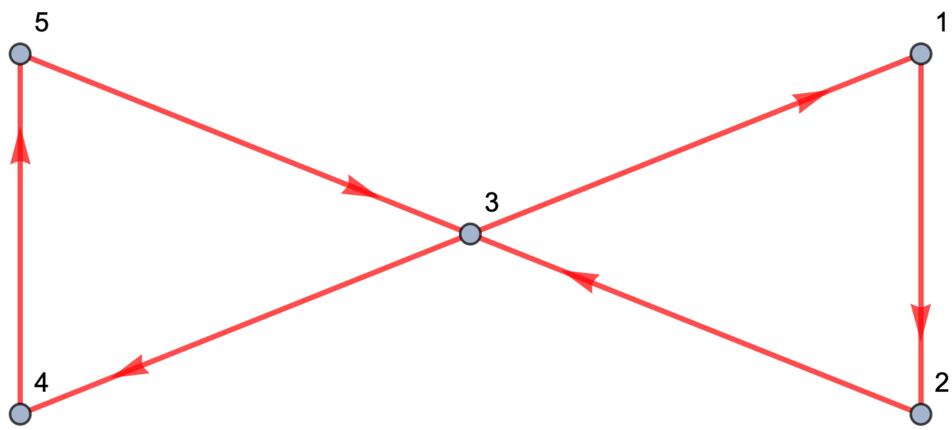
```
In[•]:= fig[Join[{3 ⟷ 4, 4 ⟷ 5, 5 ⟷ 3}, {1 ⟷ 2, 2 ⟷ 3, 3 ⟷ 1}],
         3]
```

Out[•]=




```
In[•]:= fig[{3 ⟷ 4, 4 ⟷ 5, 5 ⟷ 3, 1 ⟷ 2, 2 ⟷ 3, 3 ⟷ 1}, 3]
```

Out[•]=




```
In[•]:= HamiltonianGraphQ[gcm]
Out[•]= False

In[•]:= EulerianGraphQ[gcm]
Out[•]= True

In[•]:= FindHamiltonianCycle[gcm]
Out[•]= {}
```

In[•]:=
```
FindEulerianCycle[gcm]
```

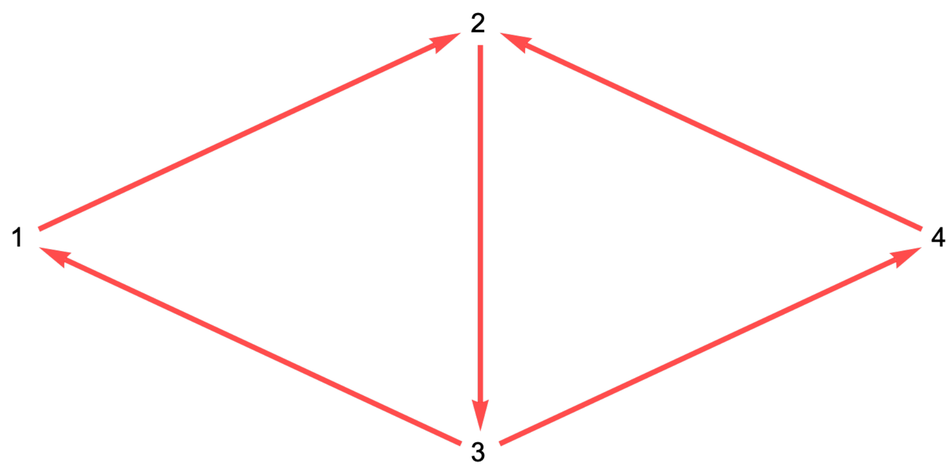
Out[•]= {{1 ↔ 2, 2 ↔ 3, 3 ↔ 4, 4 ↔ 5, 5 ↔ 3, 3 ↔ 1}}

In[•]:=
```
tm := {{0, 1, 0, 0}, {0, 0, 1, 0}, {1, 0, 0, 1},
   {0, 1, 0, 0}}
```

In[•]:=
```
gtm := makeGrph[tm, 3]
```

In[•]:=
```
gtm
```

Out[•]=



In[•]:=
```
FindEulerianCycle[gtm]
```

Out[•]= {}

In[•]:=
```
FindHamiltonianCycle[gtm]
```

Out[•]= {}

In[•]:=
```
findSomeCycles[tm, 3]
```

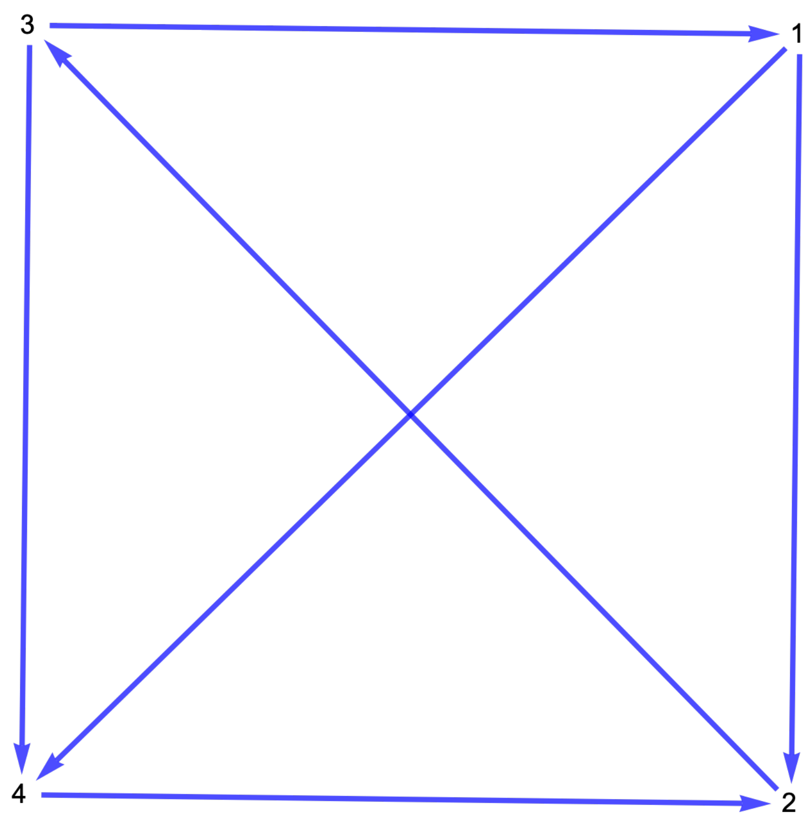
Out[•]= {{2 ↔ 3, 3 ↔ 4, 4 ↔ 2}, {1 ↔ 2, 2 ↔ 3, 3 ↔ 1}}

In[•]:=
```
M := {{0, 1, 0, 1}, {0, 0, 1, 0}, {1, 0, 0, 1},
   {0, 1, 0, 0}}
```

In[•]:=
```
gM := makeGrph[M, 2]
```

In[∘]:= gM

Out[∘]=



In[∘]:= HamiltonianGraphQ[gM]

Out[∘]= True

FindCycle[gM]

Out[∘]= {{1 ⥅ 2, 2 ⥅ 3, 3 ⥅ 1}}

In[∘]:= MatrixForm[findAllCycles[M]]

Out[∘]//MatrixForm=

$$\begin{pmatrix} \{2 \to 3,\ 3 \to 4,\ 4 \to 2\} \\ \{1 \to 2,\ 2 \to 3,\ 3 \to 1\} \\ \{1 \to 4,\ 4 \to 2,\ 2 \to 3,\ 3 \to 1\} \end{pmatrix}$$

```
In[∘]:= sM := {{0, 1, 1, 1}, {1, 0, 1, 1}, {1, 0, 0, 1},
   {1, 1, 1, 0}}
```

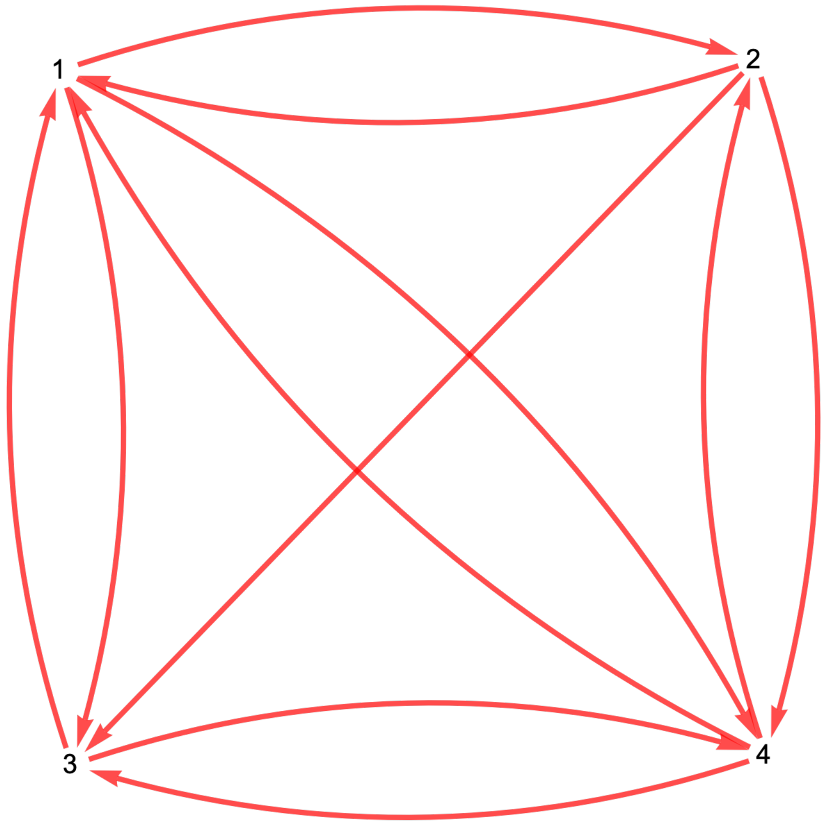
```
In[•]:= MatrixForm[findAllCycles[sM]]

Out[•]//MatrixForm=
```

$$
\begin{pmatrix}
\{3 \mapsto 4, 4 \mapsto 3\} \\
\{2 \mapsto 4, 4 \mapsto 2\} \\
\{1 \mapsto 4, 4 \mapsto 1\} \\
\{1 \mapsto 3, 3 \mapsto 1\} \\
\{1 \mapsto 2, 2 \mapsto 1\} \\
\{2 \mapsto 3, 3 \mapsto 4, 4 \mapsto 2\} \\
\{1 \mapsto 4, 4 \mapsto 3, 3 \mapsto 1\} \\
\{1 \mapsto 4, 4 \mapsto 2, 2 \mapsto 1\} \\
\{1 \mapsto 3, 3 \mapsto 4, 4 \mapsto 1\} \\
\{1 \mapsto 2, 2 \mapsto 4, 4 \mapsto 1\} \\
\{1 \mapsto 2, 2 \mapsto 3, 3 \mapsto 1\} \\
\{1 \mapsto 4, 4 \mapsto 2, 2 \mapsto 3, 3 \mapsto 1\} \\
\{1 \mapsto 3, 3 \mapsto 4, 4 \mapsto 2, 2 \mapsto 1\} \\
\{1 \mapsto 2, 2 \mapsto 4, 4 \mapsto 3, 3 \mapsto 1\} \\
\{1 \mapsto 2, 2 \mapsto 3, 3 \mapsto 4, 4 \mapsto 1\}
\end{pmatrix}
$$

In[∘]:= makeGrph[sM, 3]

Out[∘]=



In[∘]:= eM := {{0, 1, 1, 1}, {1, 0, 1, 1}, {1, 1, 0, 1},
{1, 1, 1, 0}}

In[∘]:= EulerianGraphQ[eM]

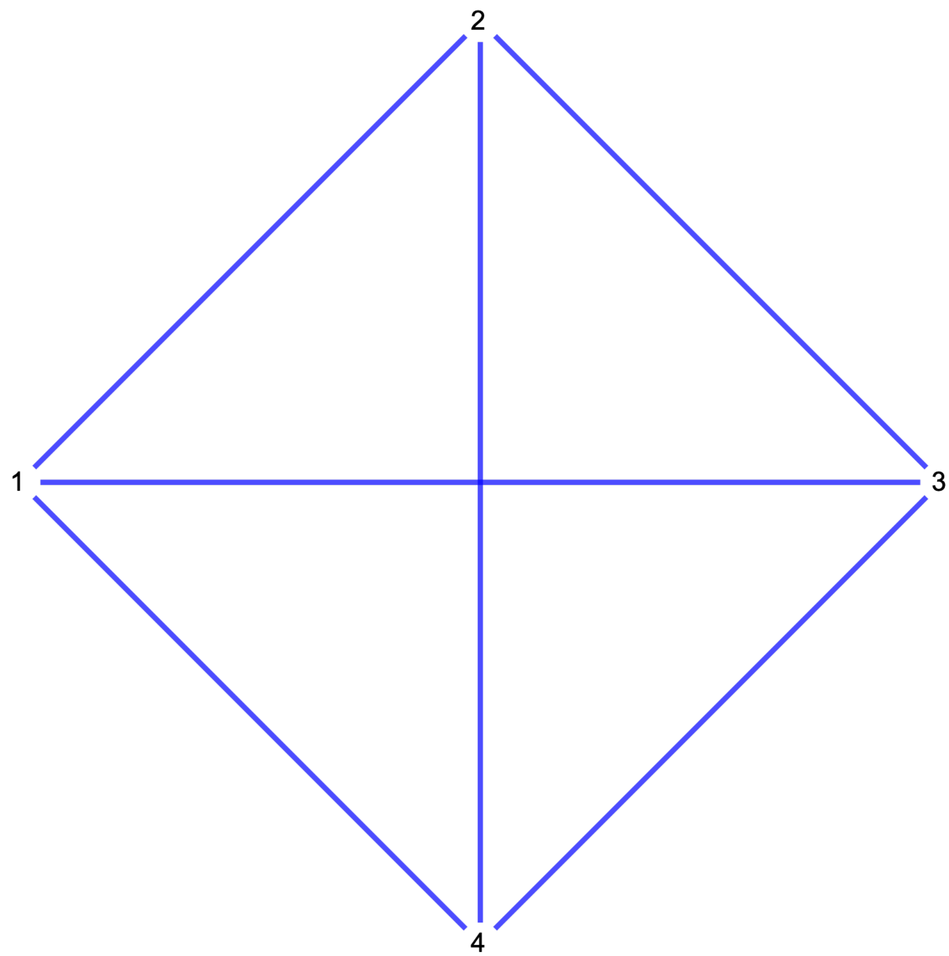
Out[∘]= False

In[∘]:= HamiltonianGraphQ[eM]

Out[∘]= False

In[∘]:= `makeGrph[eM, 2]`

Out[∘]=



In[∘]:= `MatrixForm[findAllCycles[eM]]`

Out[∘]//MatrixForm=

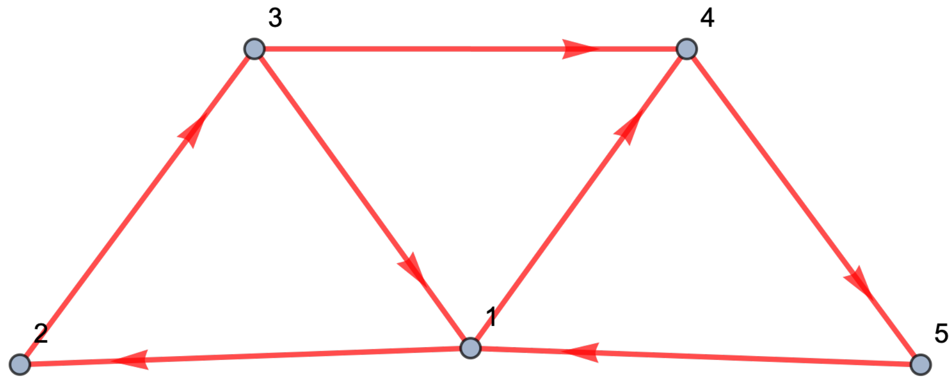
$$\begin{pmatrix} \{2 \leftrightarrow 3, 3 \leftrightarrow 4, 4 \leftrightarrow 2\} \\ \{1 \leftrightarrow 4, 4 \leftrightarrow 2, 2 \leftrightarrow 1\} \\ \{1 \leftrightarrow 2, 2 \leftrightarrow 3, 3 \leftrightarrow 1\} \\ \{1 \leftrightarrow 4, 4 \leftrightarrow 3, 3 \leftrightarrow 1\} \\ \{1 \leftrightarrow 4, 4 \leftrightarrow 3, 3 \leftrightarrow 2, 2 \leftrightarrow 1\} \\ \{1 \leftrightarrow 2, 2 \leftrightarrow 4, 4 \leftrightarrow 3, 3 \leftrightarrow 1\} \\ \{1 \leftrightarrow 4, 4 \leftrightarrow 2, 2 \leftrightarrow 3, 3 \leftrightarrow 1\} \end{pmatrix}$$

eM has Hamiltonian sub-circuits but it is not a Hamiltonian circuit.

```
In[•]:= tl := {1 ↦ 2, 2 ↦ 3, 3 ↦ 1, 1 ↦ 4, 4 ↦ 5, 5 ↦ 1, 3 ↦ 4}

In[•]:= fig[tl, 3]
```

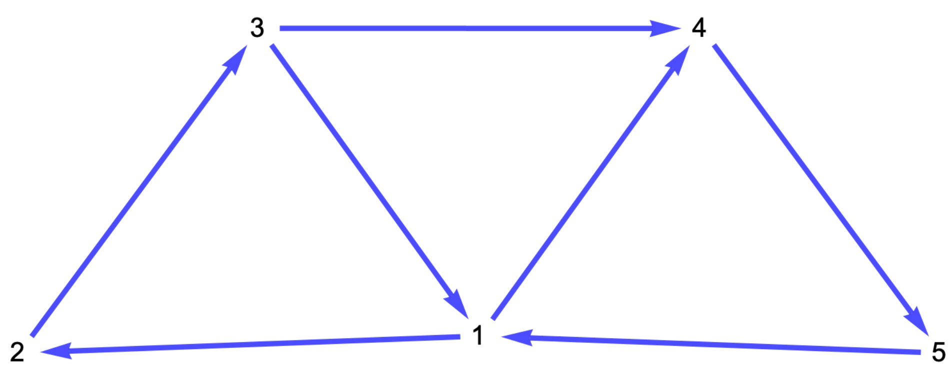
Out[•]=




```
In[•]:= tlm := {{0, 1, 0, 1, 0}, {0, 0, 1, 0, 0}, {1, 0, 0, 1, 0},
           {0, 0, 0, 0, 1}, {1, 0, 0, 0, 0}}

In[•]:= makeGrph[tlm, 2]
```

Out[•]=




```
In[•]:= MatrixForm[findAllCycles[tlm]]
```

Out[•]//MatrixForm=

$$\begin{pmatrix} \{1 \mapsto 4,\ 4 \mapsto 5,\ 5 \mapsto 1\} \\ \{1 \mapsto 2,\ 2 \mapsto 3,\ 3 \mapsto 1\} \\ \{1 \mapsto 2,\ 2 \mapsto 3,\ 3 \mapsto 4,\ 4 \mapsto 5,\ 5 \mapsto 1\} \end{pmatrix}$$

```
In[•]:= HamiltonianGraphQ[tlm]

Out[•]= False
```

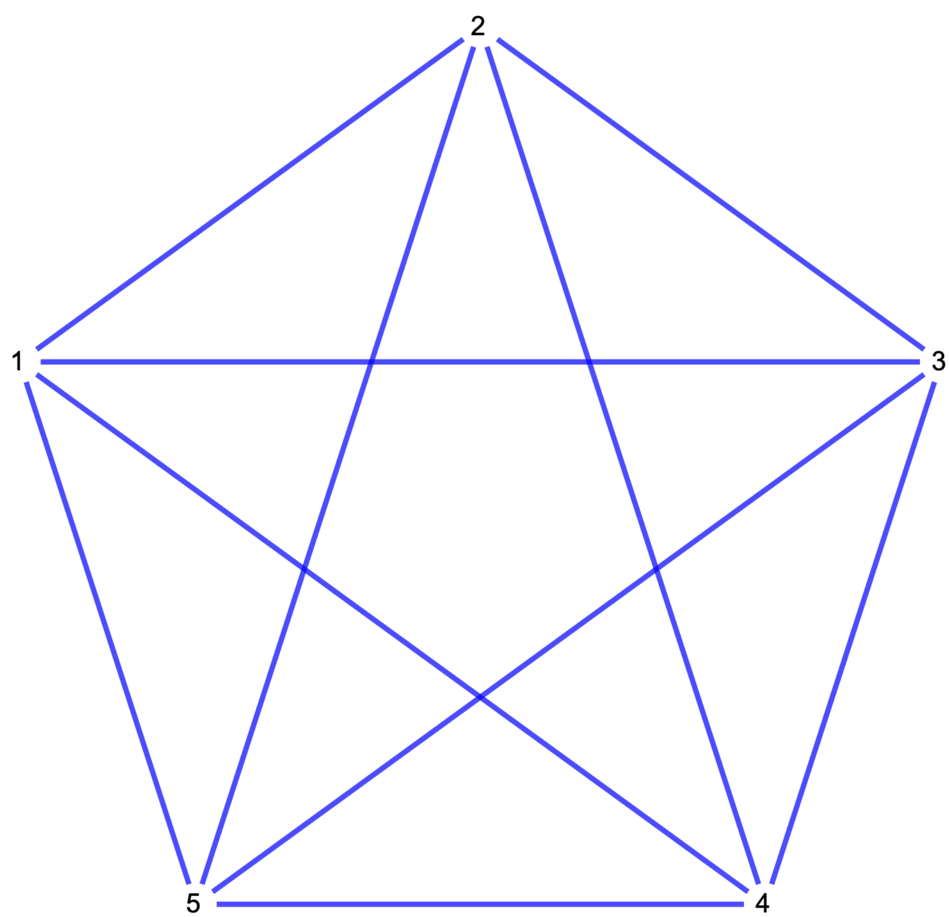
Some vertexes are odd.

```
In[•]:= s5 := {{0, 1, 1, 1, 1}, {1, 0, 1, 1, 1}, {1, 1, 0, 1, 1},
   {1, 1, 1, 0, 1}, {1, 1, 1, 1, 0}}

In[•]:= gs5 := makeGrph[s5, 2]

In[•]:= gs5
```

Out[•]=




```
In[•]:= EulerianGraphQ[gs5]

Out[•]= True
```

The symmetric square has no Eulerian graph because its vertexes are odd. The symmetric pentagon has an Eulerian graph because its vertexes are even.

References

Bondareva, O . N . 1963. Some Applications of Linear Programming Methods to the Theory of Cooperative Games (in Russian). Problemy Kibernetiki 10 : 119 - 39.

Programs