Graph Threading

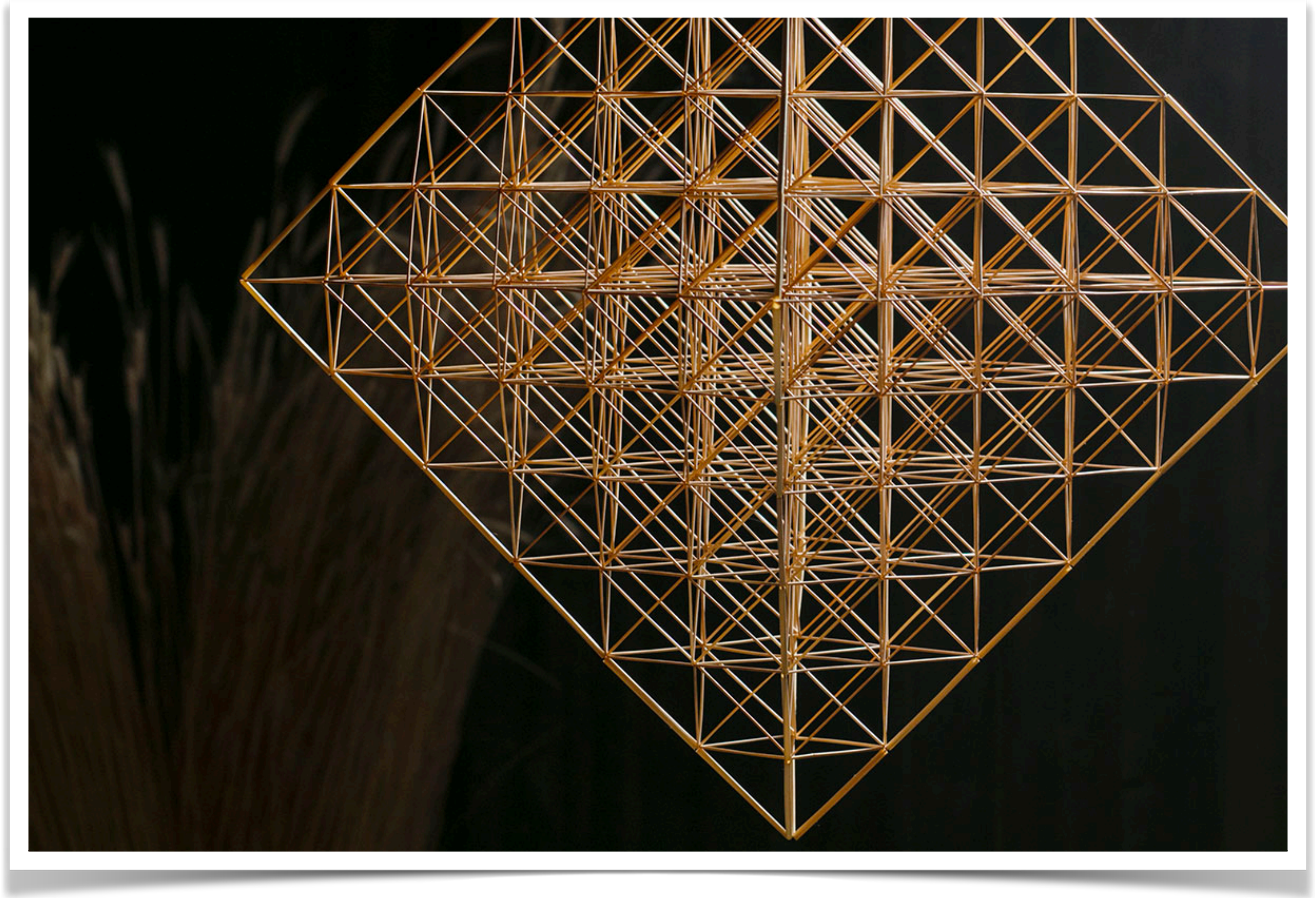
Erik Demaine, Yael Kirkpatrick, and Rebecca Lin

ITCS 2024

Massachusetts Institute of Technology

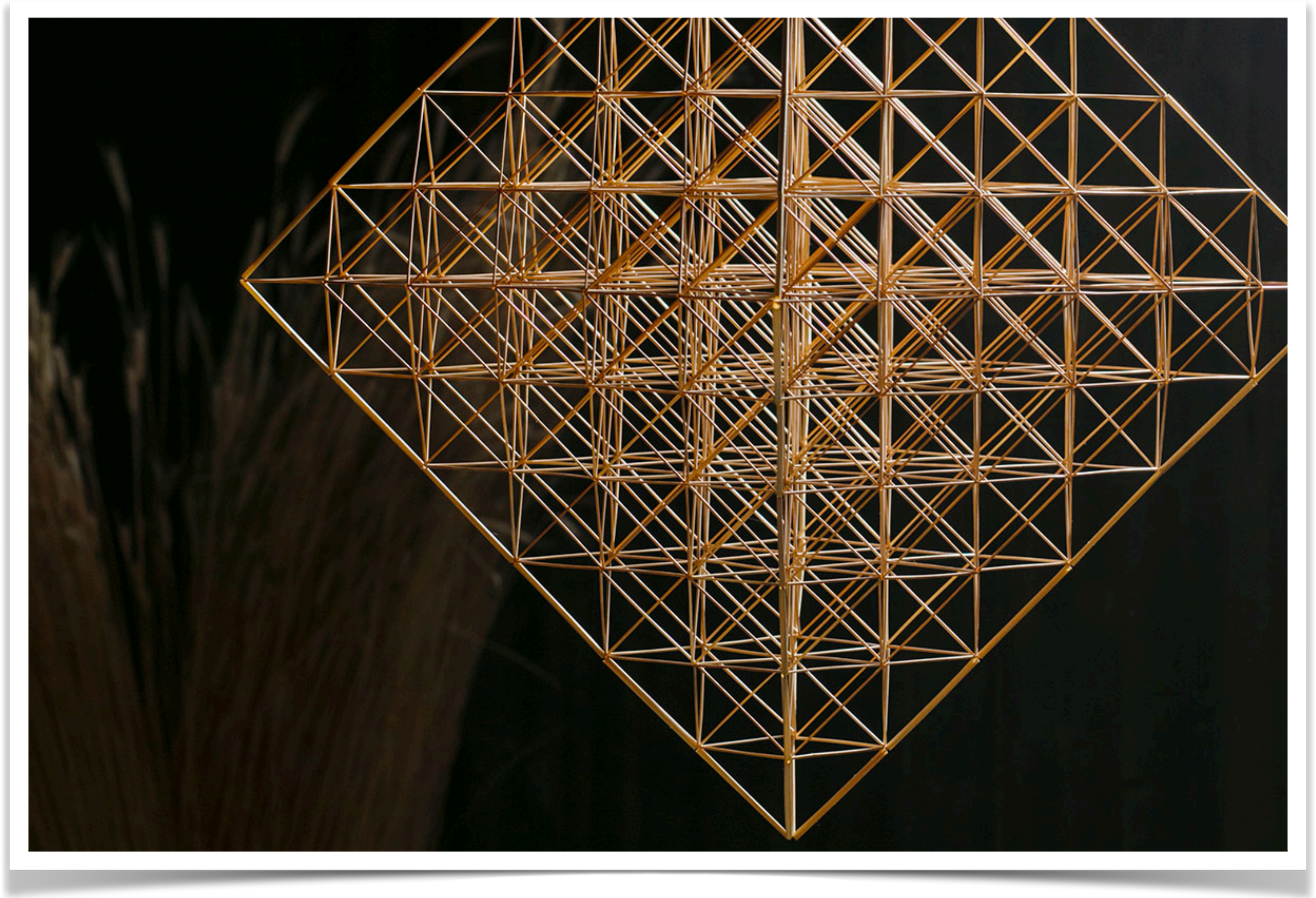
beadwork from Beady
[Igarashi et al. 2012]

beadwork from Beady
[Igarashi et al. 2012]



a himmeli by Eija Koski

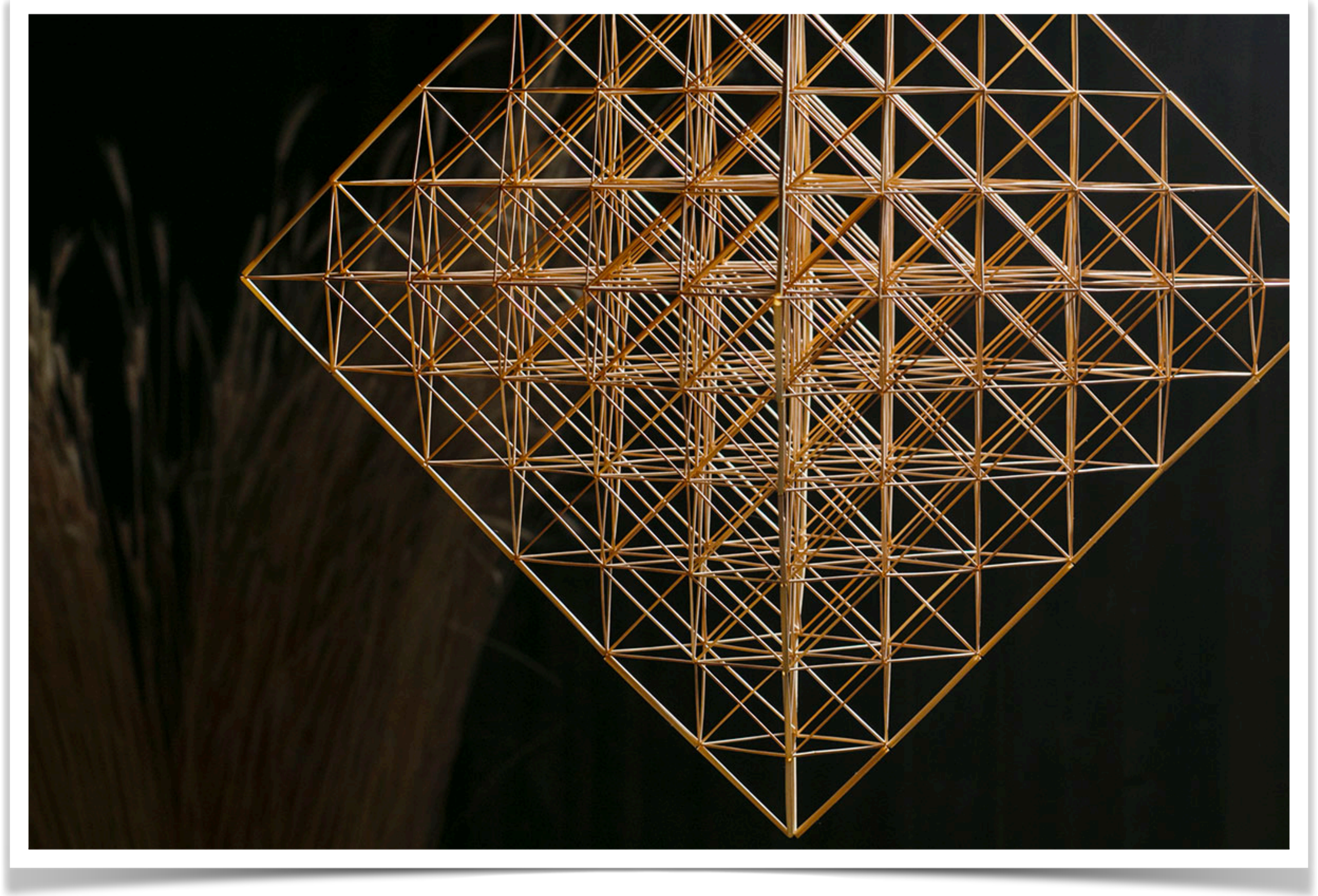
beadwork from Beady
[Igarashi et al. 2012]



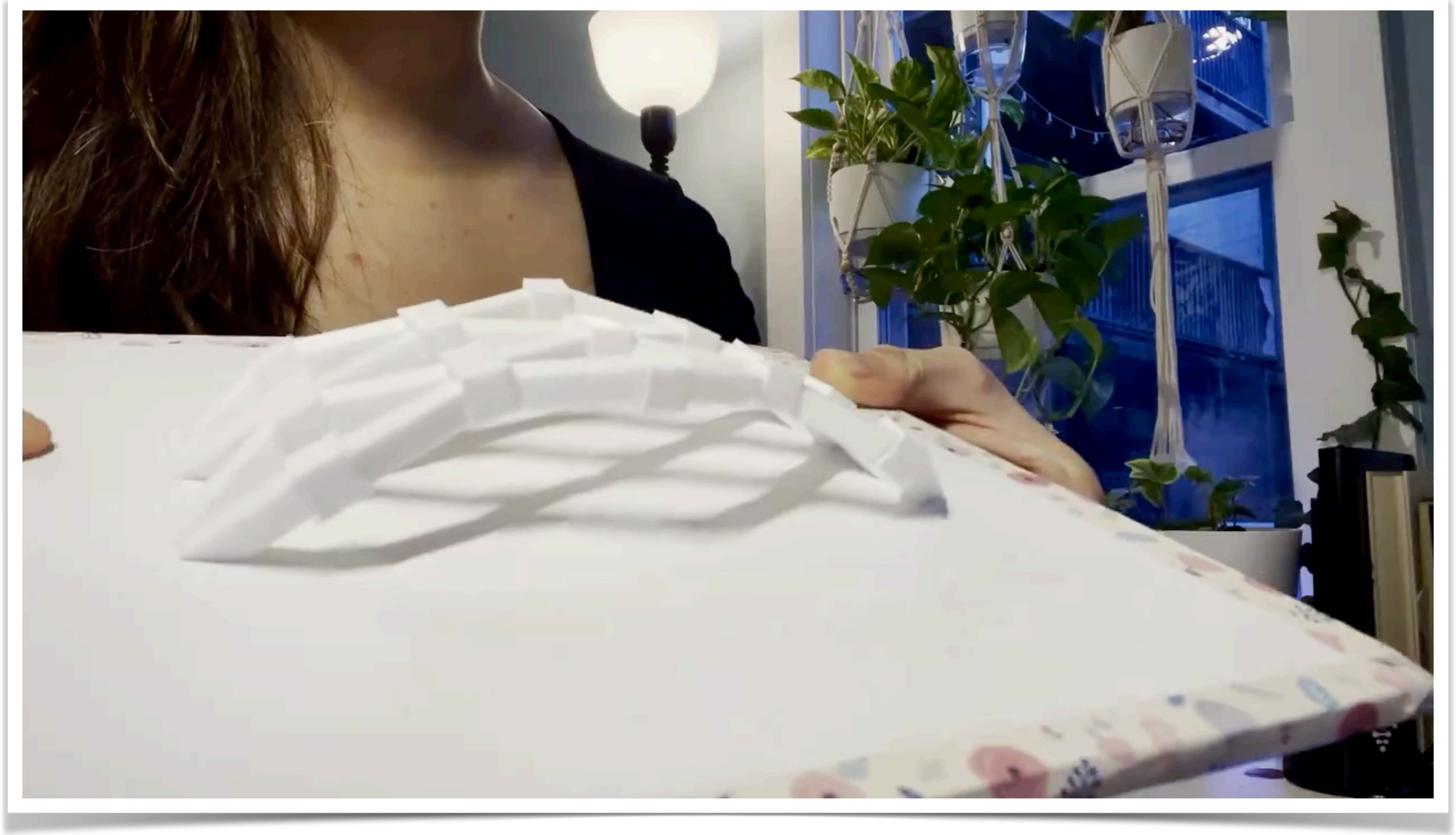
a himmeli by Eija Koski

a deployable dome by Alison Martin

beadwork from Beady
[Igarashi et al. 2012]

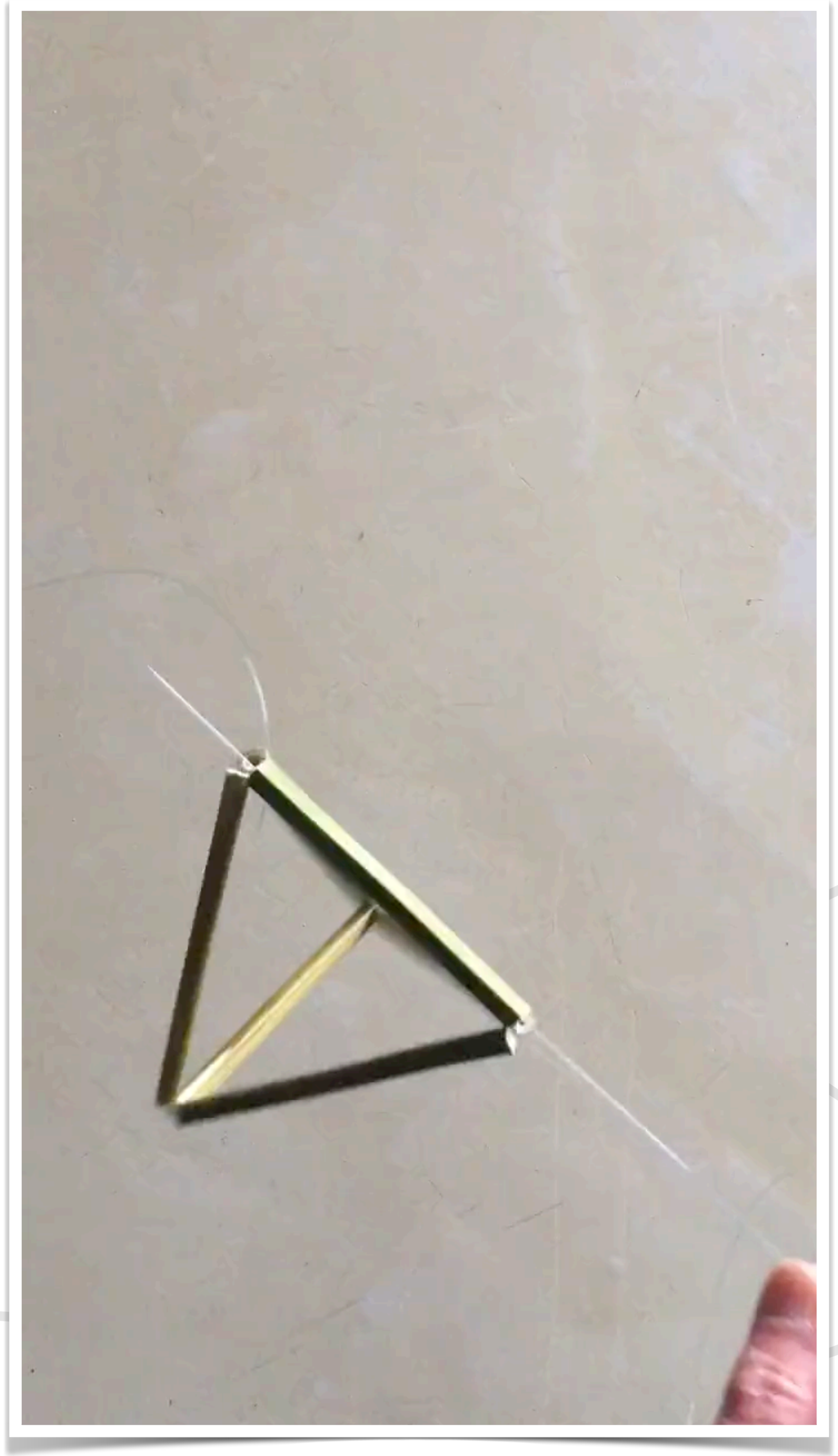


a himmeli by Eija Koski



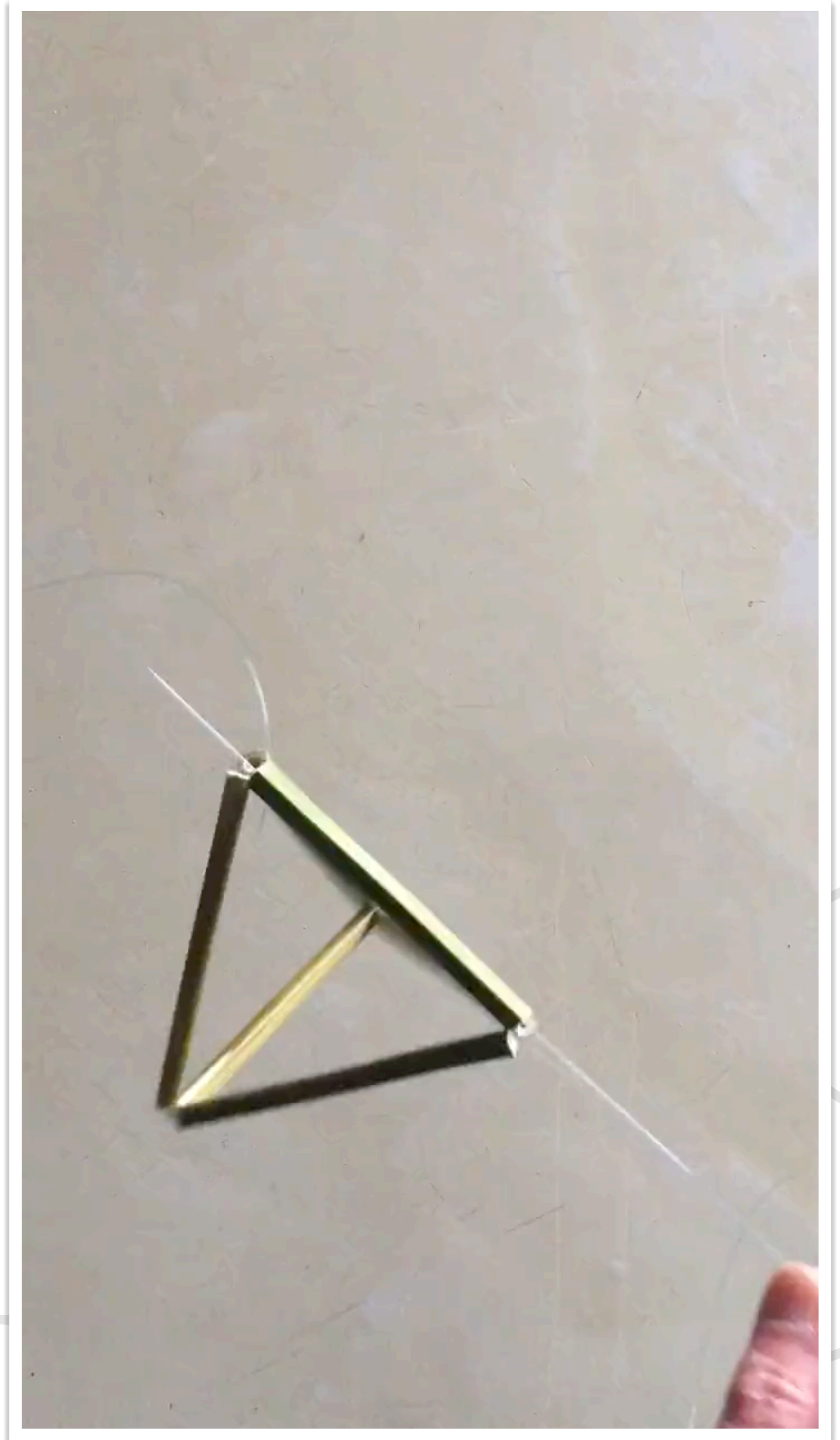
a push puppet-inspired deployable structure by Lin and Tachi

a deployable dome by Alison Martin



a tetrahedron by Alison Martin

How do we efficiently **thread** a string through a collection of tubes so that the tubes connect as intended when the string is pulled taut?

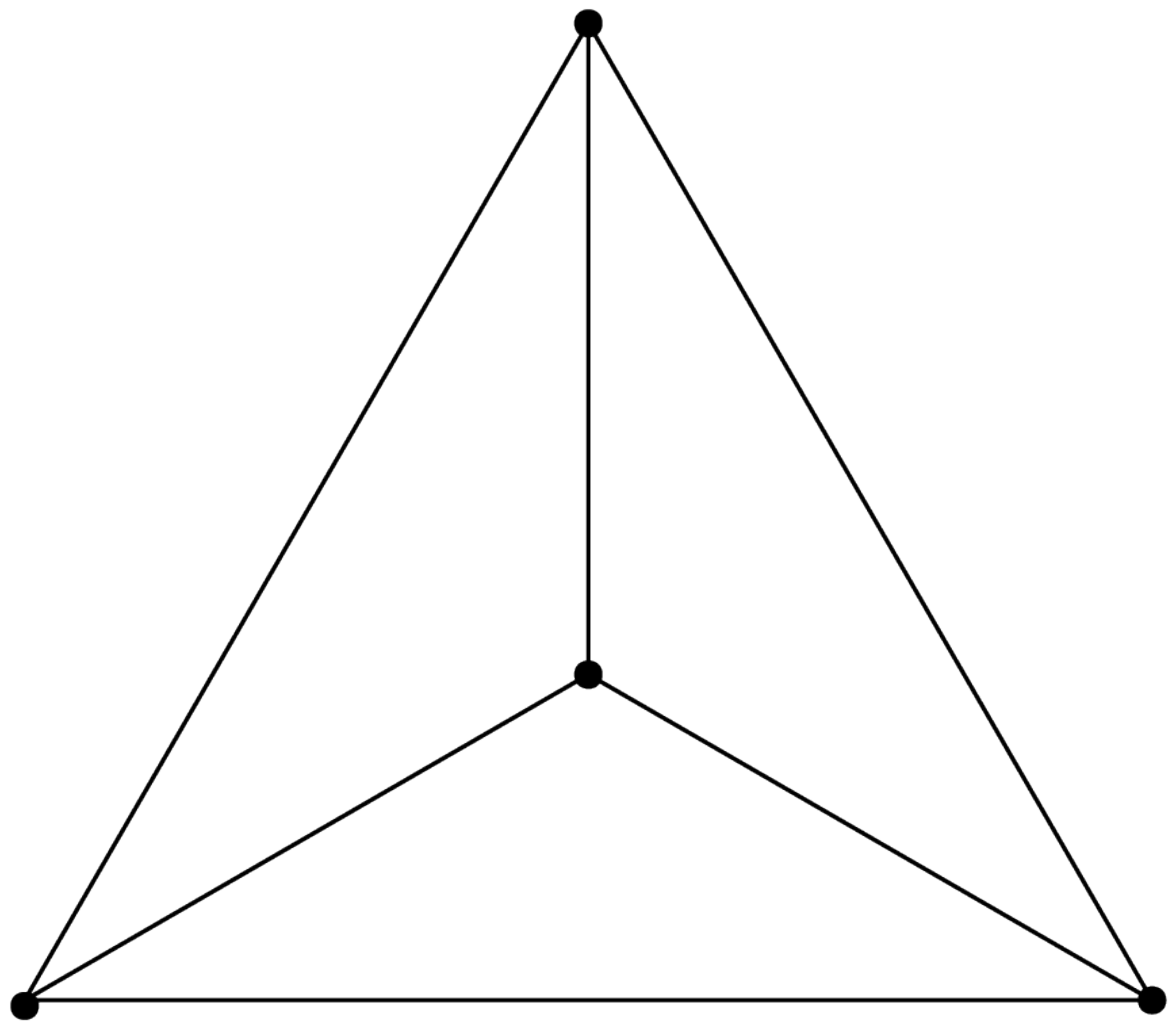


a tetrahedron by Alison Martin

How do we efficiently **thread** a string through a collection of tubes so that the tubes connect as intended when the string is pulled taut?

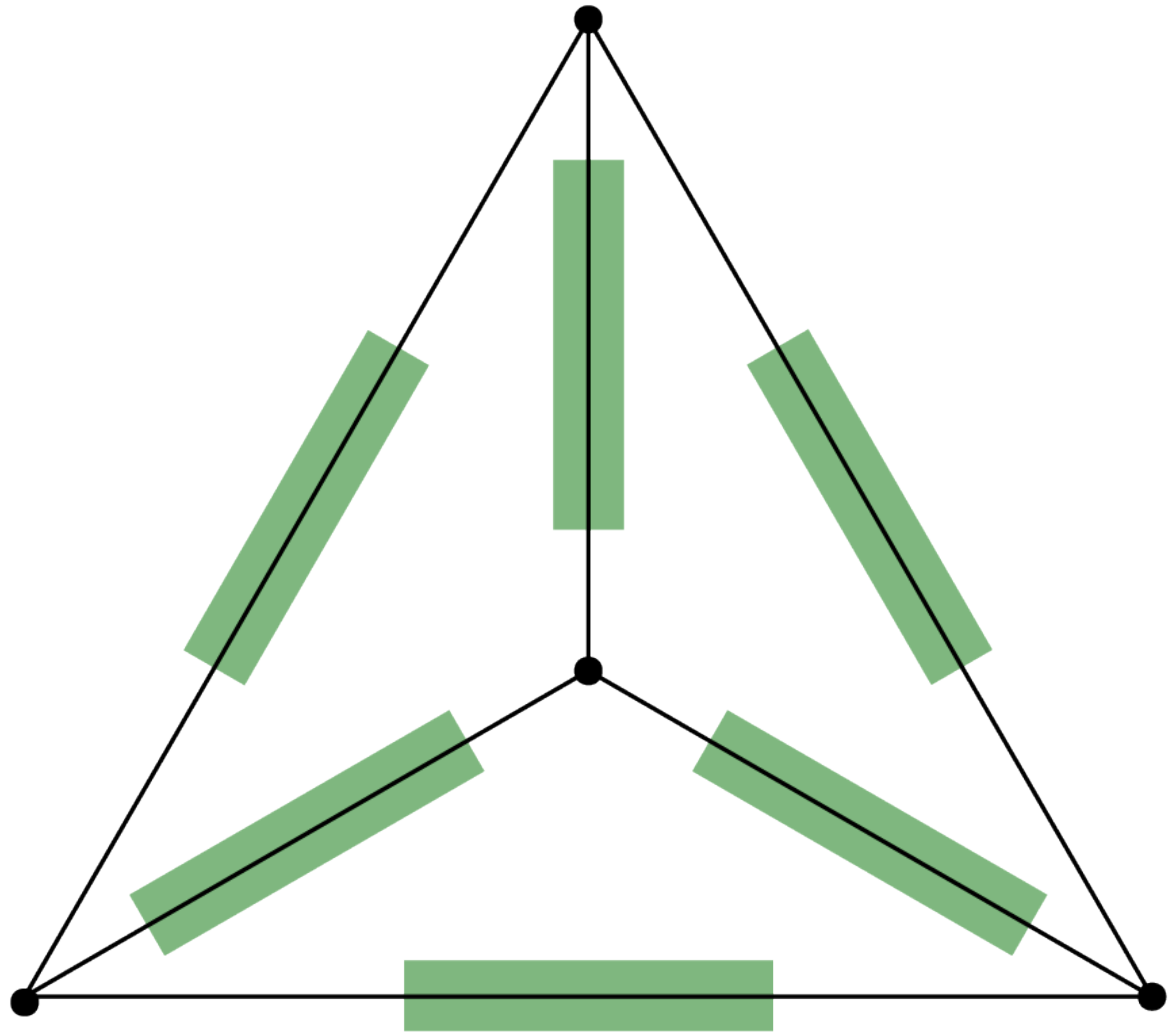
The Threading Problem

The Threading Problem



$G = (V, E)$

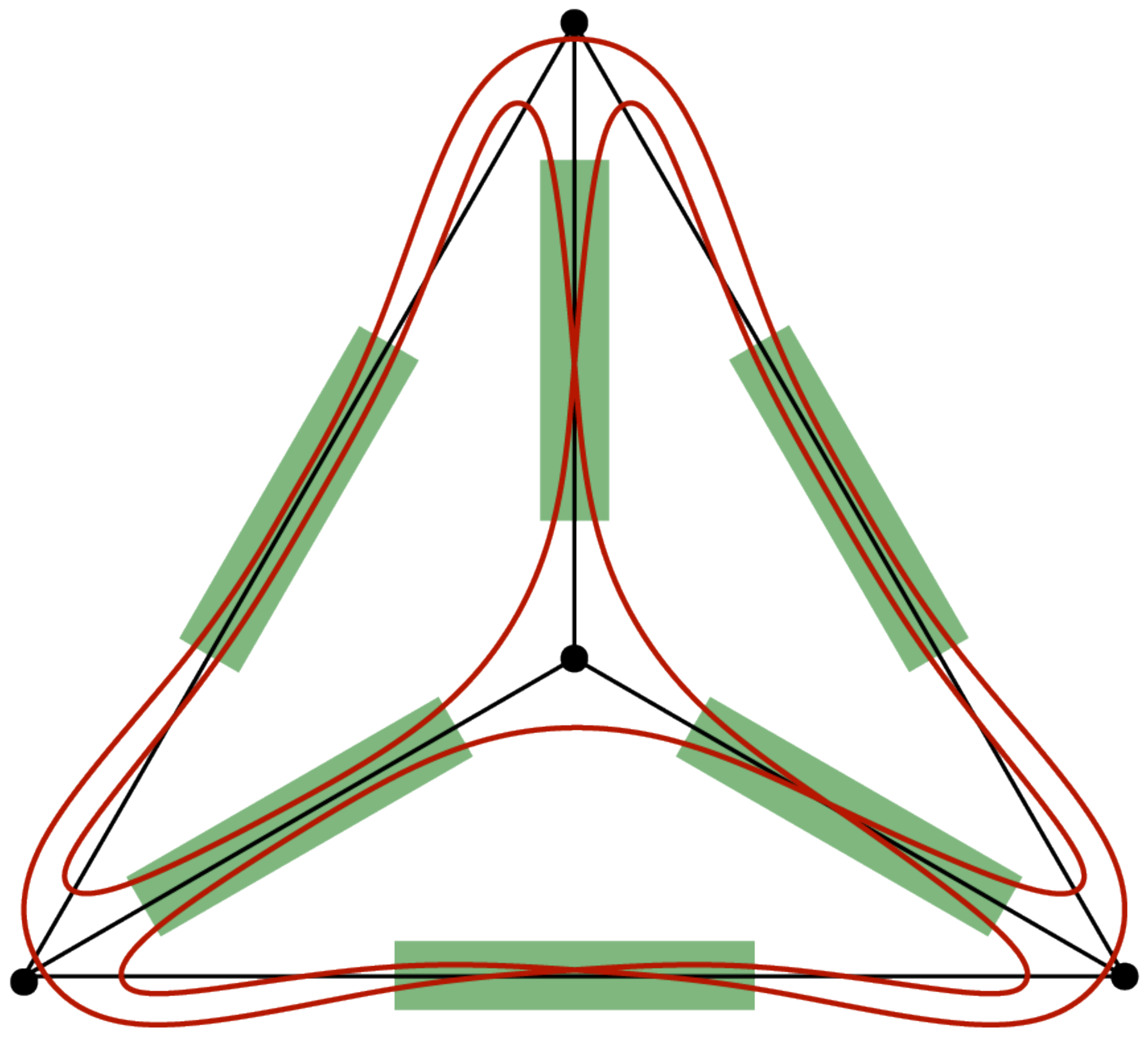
The Threading Problem



$G = (V, E)$

The Threading Problem

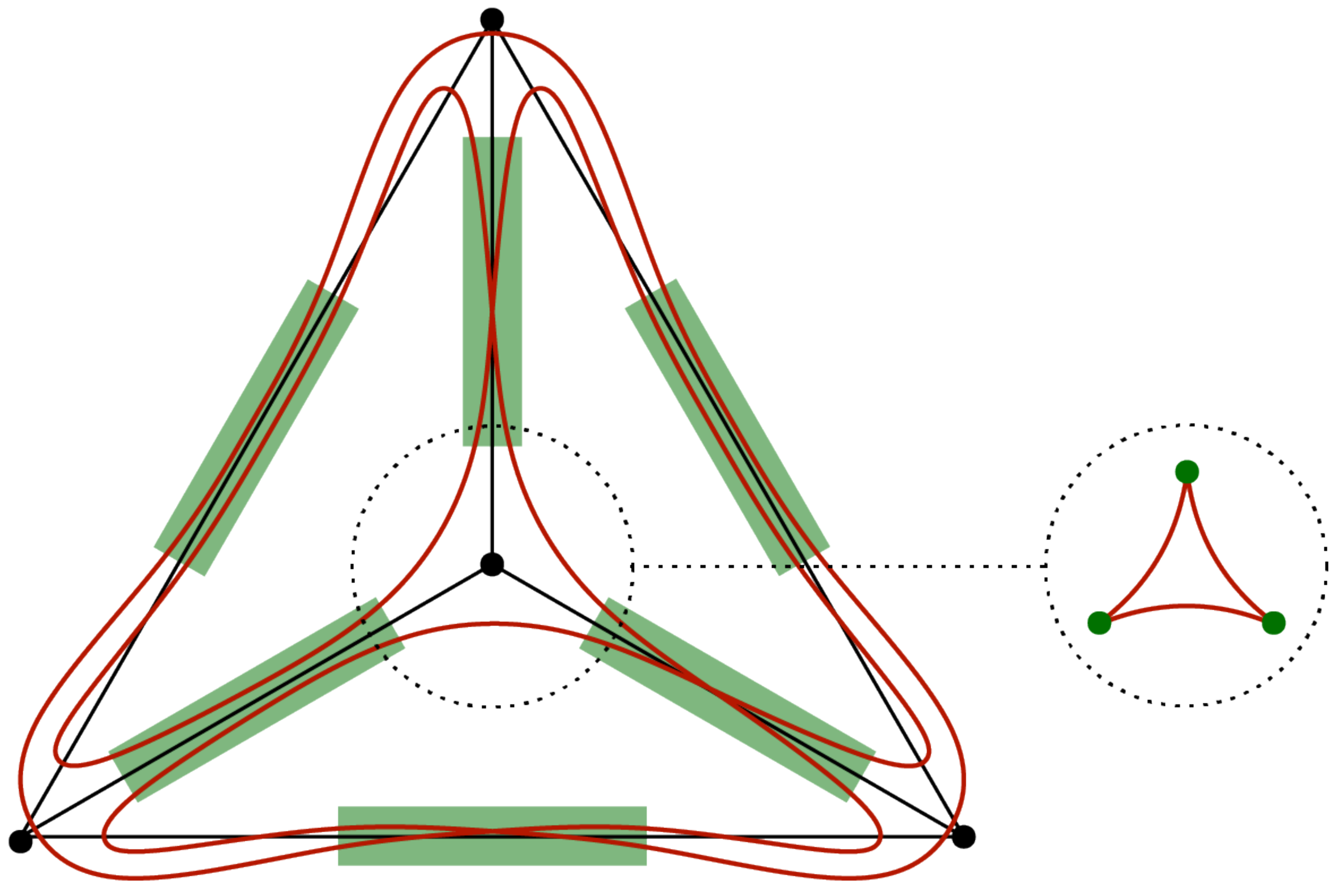
A **threading** T of G is a closed walk through G that visits each edge at least once,



$G = (V, E)$

The Threading Problem

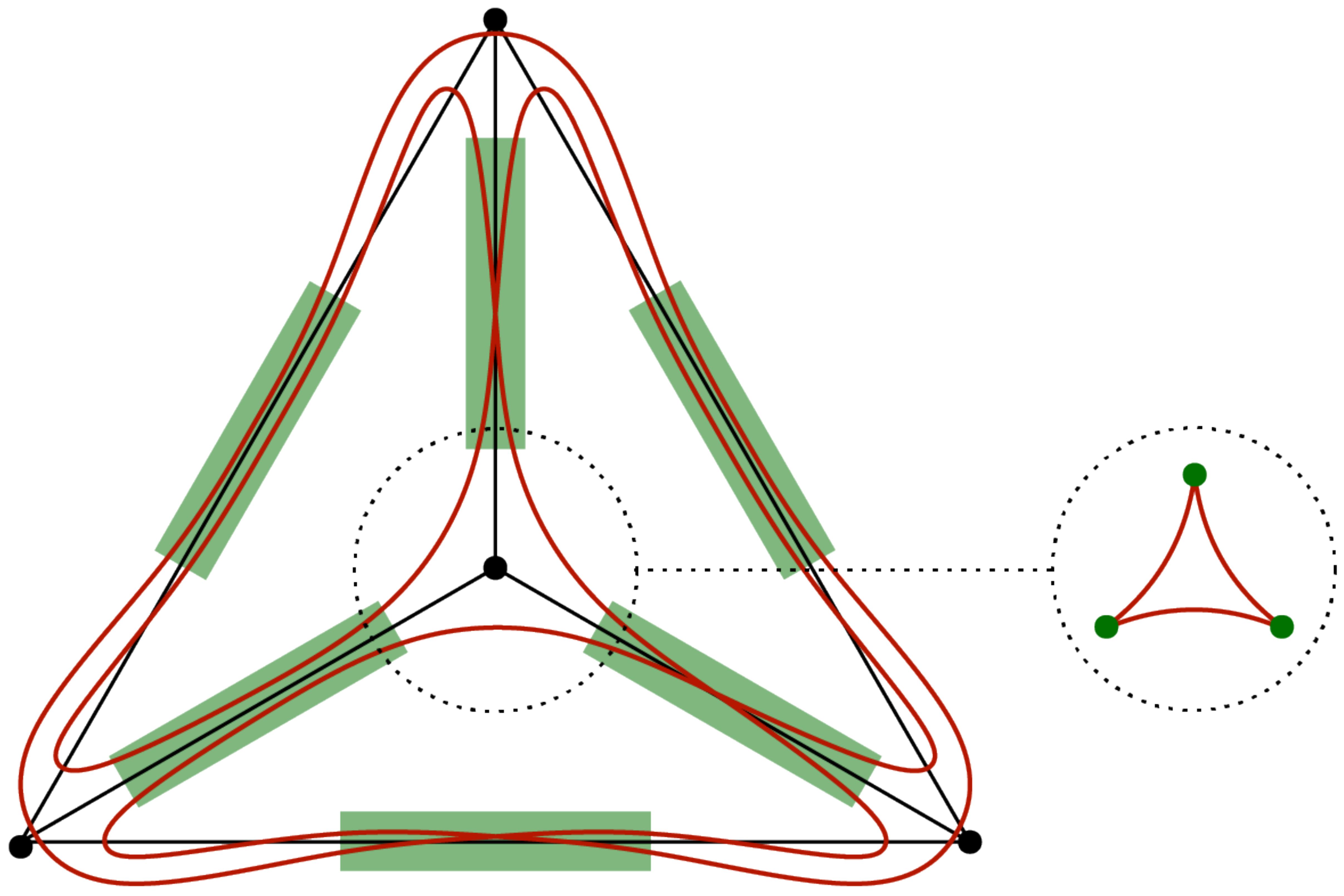
A **threading** T of G is a closed walk through G that visits each edge at least once,
induces connected "junction graphs" at each vertex,



$G = (V, E)$

The Threading Problem

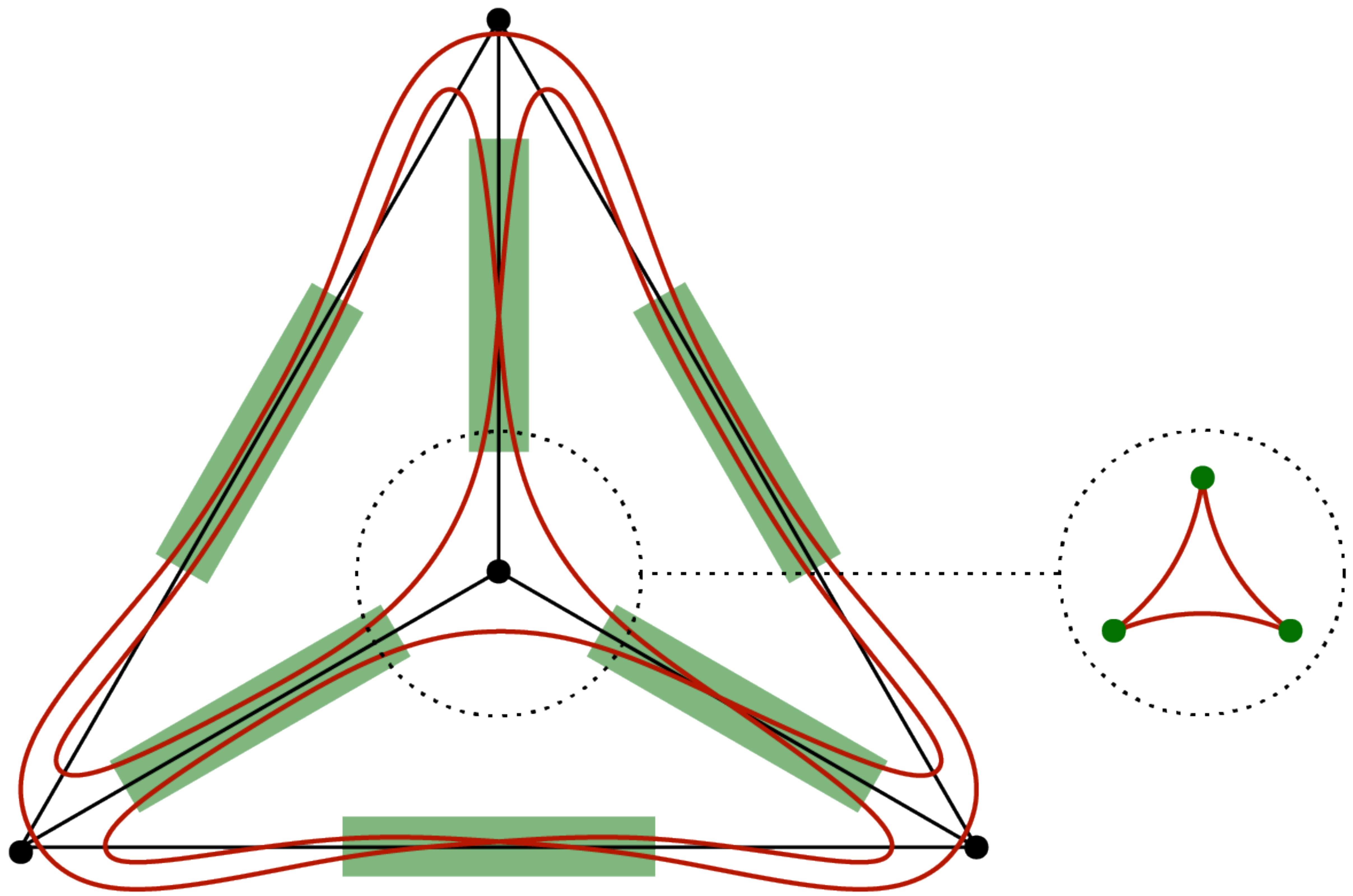
A **threading** T of G is a closed walk through G that visits each edge at least once, induces connected "junction graphs" at each vertex, and has no "U-turns".



$G = (V, E)$

The Threading Problem

A **threading** T of G is a closed walk through G that visits each edge at least once, induces connected "junction graphs" at each vertex, and has no "U-turns".

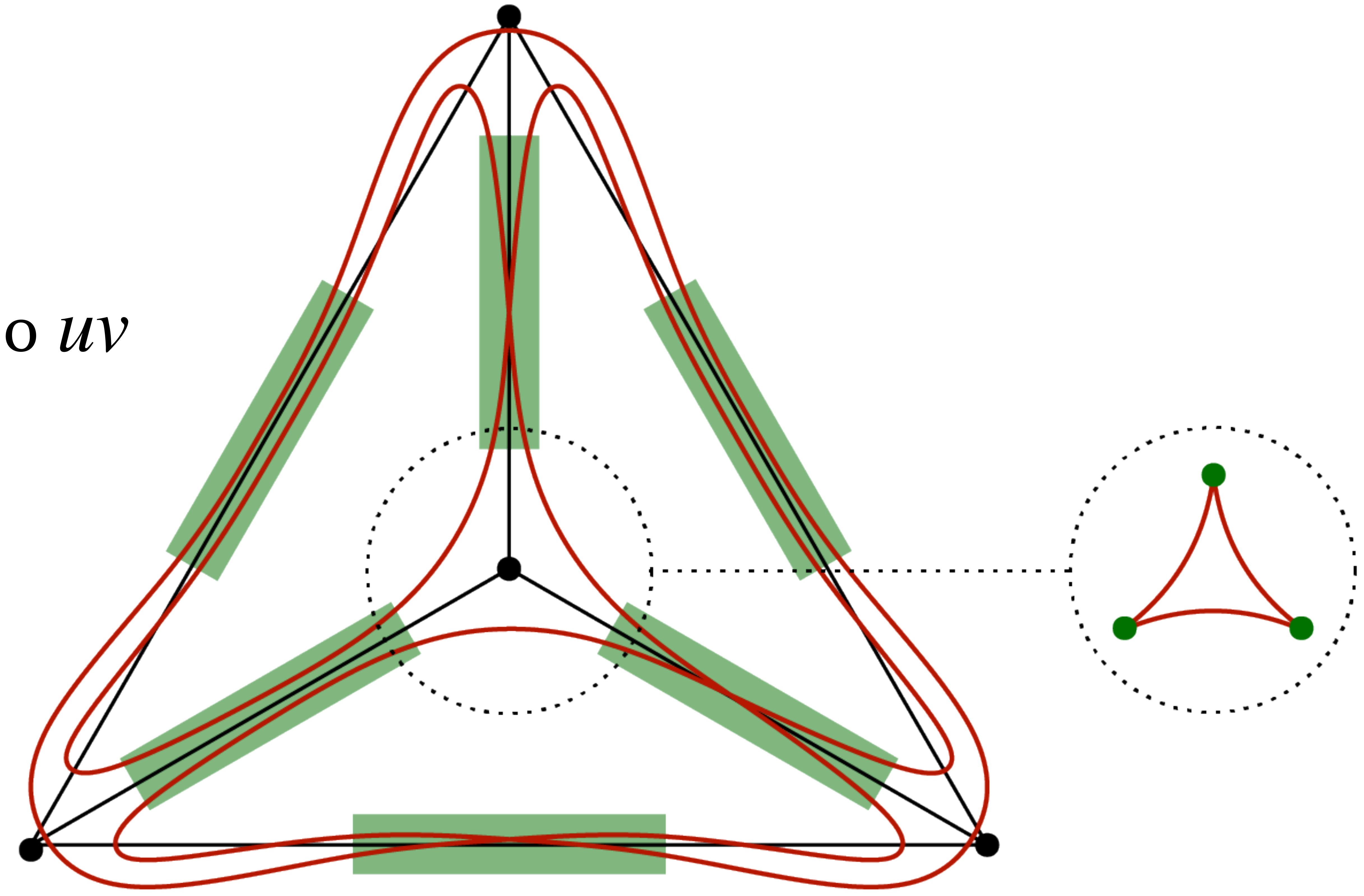


$G = (V, E)$

The Threading Problem

A **threading** T of G is a closed walk through G that visits each edge at least once, induces connected "junction graphs" at each vertex, and has no "U-turns".

count $x_{uv} =$ # of visits to uv



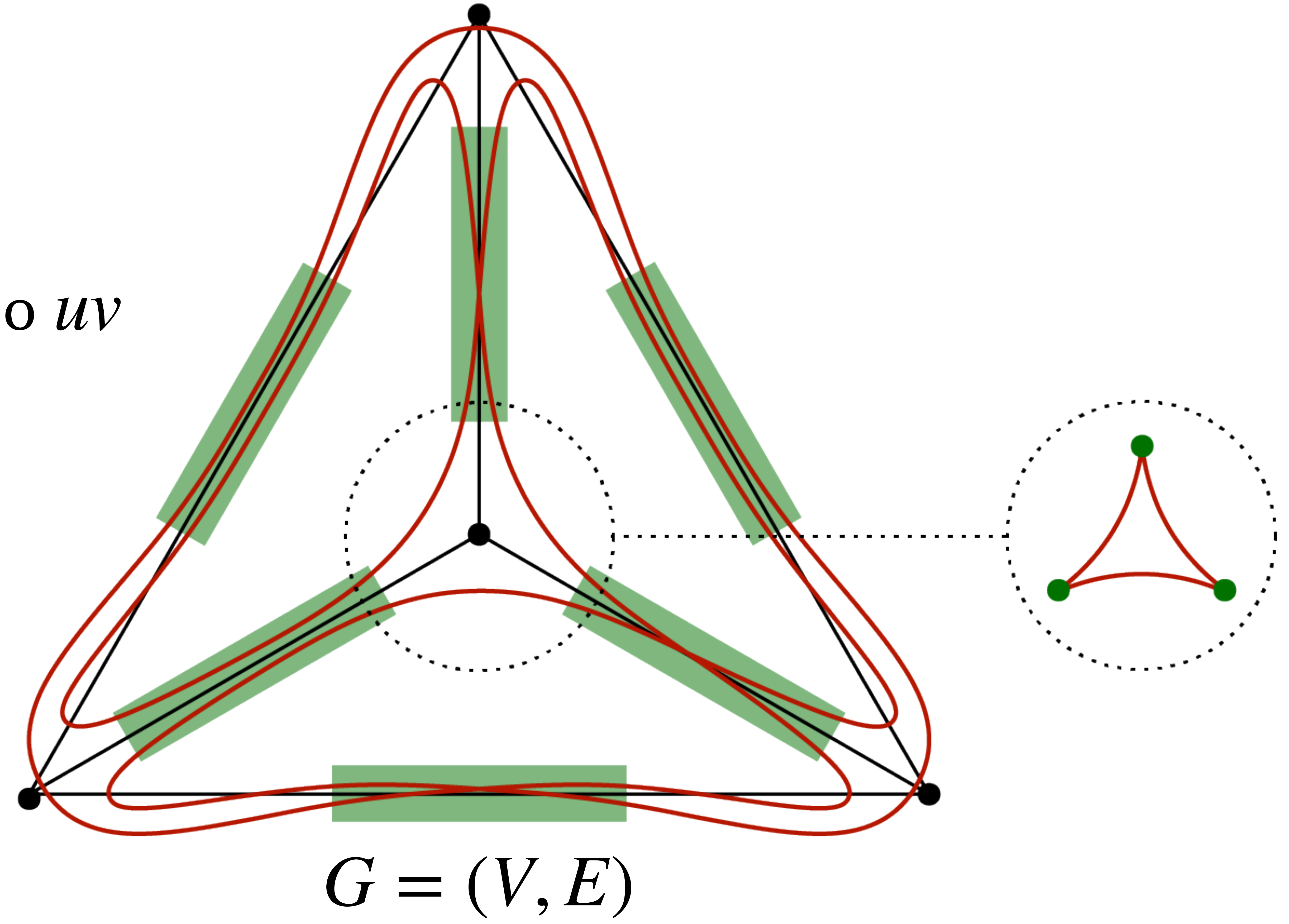
$G = (V, E)$

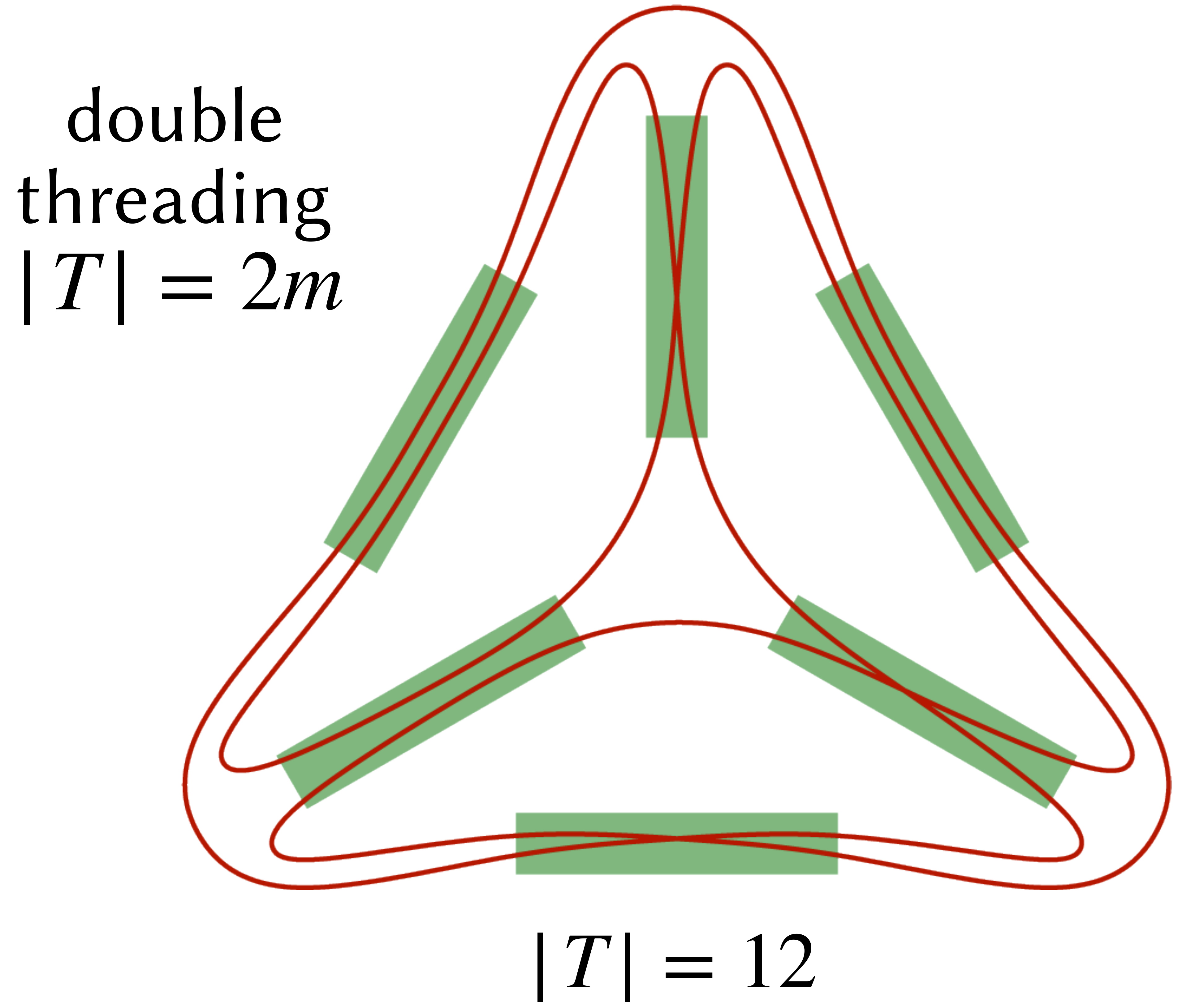
The Threading Problem

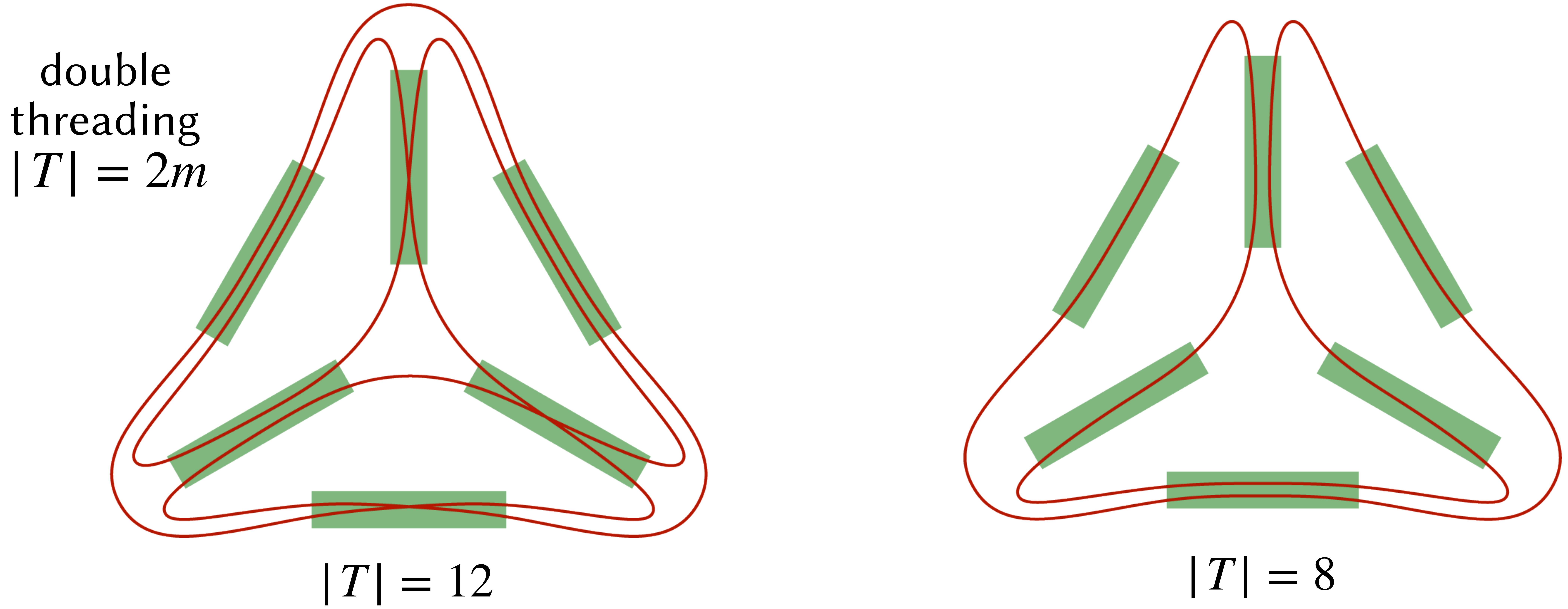
A **threading** T of G is a closed walk through G that visits each edge at least once, induces connected "junction graphs" at each vertex, and has no "U-turns".

count $\quad x_{uv} =$ # of visits to uv

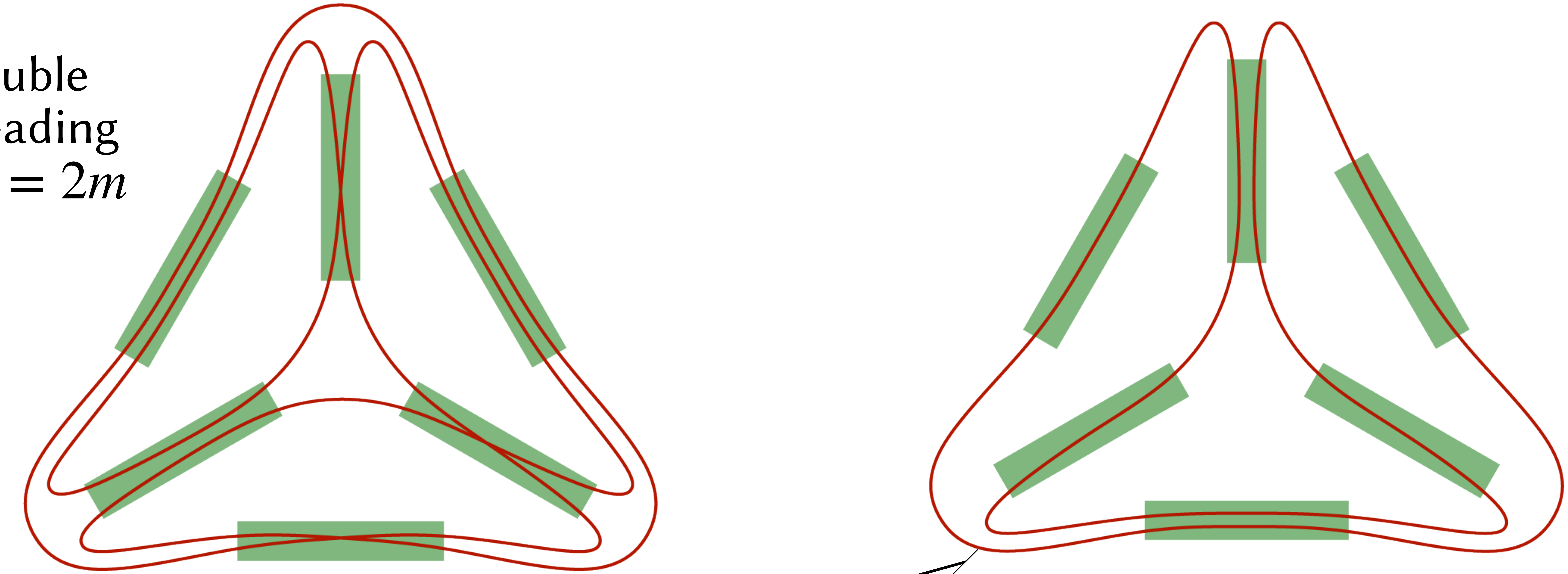
length $|T| = \sum_{uv \in E} x_{uv}$




double
threading
$|T| = 2m$
$|T| = 12$


double
threading
$|T| = 2m$
$|T| = 12$
$|T| = 8$

double threading
$|T| = 2m$



$|T| = 12$ $|T| = 8$

Goal: Minimize $|T| = \sum_{uv \in E} x_{uv}$

Our Results

Our Results

Algorithms:

Our Results

Algorithms:

- Polynomial-time algorithm to compute the edge counts of an optimal threading

Our Results

Algorithms:

- Polynomial-time algorithm to compute the edge counts of an optimal threading
- Polynomial-time algorithm to construct an optimal threading from edge counts

Our Results

Algorithms:

- Polynomial-time algorithm to compute the edge counts of an optimal threading
- Polynomial-time algorithm to construct an optimal threading from edge counts
- Improved algorithms for special cases

Our Results

Algorithms:

- Polynomial-time algorithm to compute the edge counts of an optimal threading
- Polynomial-time algorithm to construct an optimal threading from edge counts
- Improved algorithms for special cases

Bounds:

Our Results

Algorithms:

- Polynomial-time algorithm to compute the edge counts of an optimal threading
- Polynomial-time algorithm to construct an optimal threading from edge counts
- Improved algorithms for special cases

Bounds:

- An optimal threading has length at least $2m - n$

Our Results

Algorithms:

- Polynomial-time algorithm to compute the edge counts of an optimal threading
- Polynomial-time algorithm to construct an optimal threading from edge counts
- Improved algorithms for special cases

Bounds:

- An optimal threading has length at least $2m - n$
- There exist graphs with optimal threadings of length $2m - O(1)$

Overview of Talk

Overview of Talk

- Reformulate the Threading problem in terms of **local constraints** of a threading

Overview of Talk

- Reformulate the Threading problem in terms of **local constraints** of a threading
- Show a linear-time algorithm to **construct a global threading** from local solutions

Overview of Talk

- Reformulate the Threading problem in terms of **local constraints** of a threading
- Show a linear-time algorithm to **construct a global threading** from local solutions
- Give an algorithm to **compute local solutions** via reduction to perfect matching

Local Constraints of a Threading

Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod 2$ for all $v \in V$

entering *and* exiting each vertex

Local Constraints of a Threading

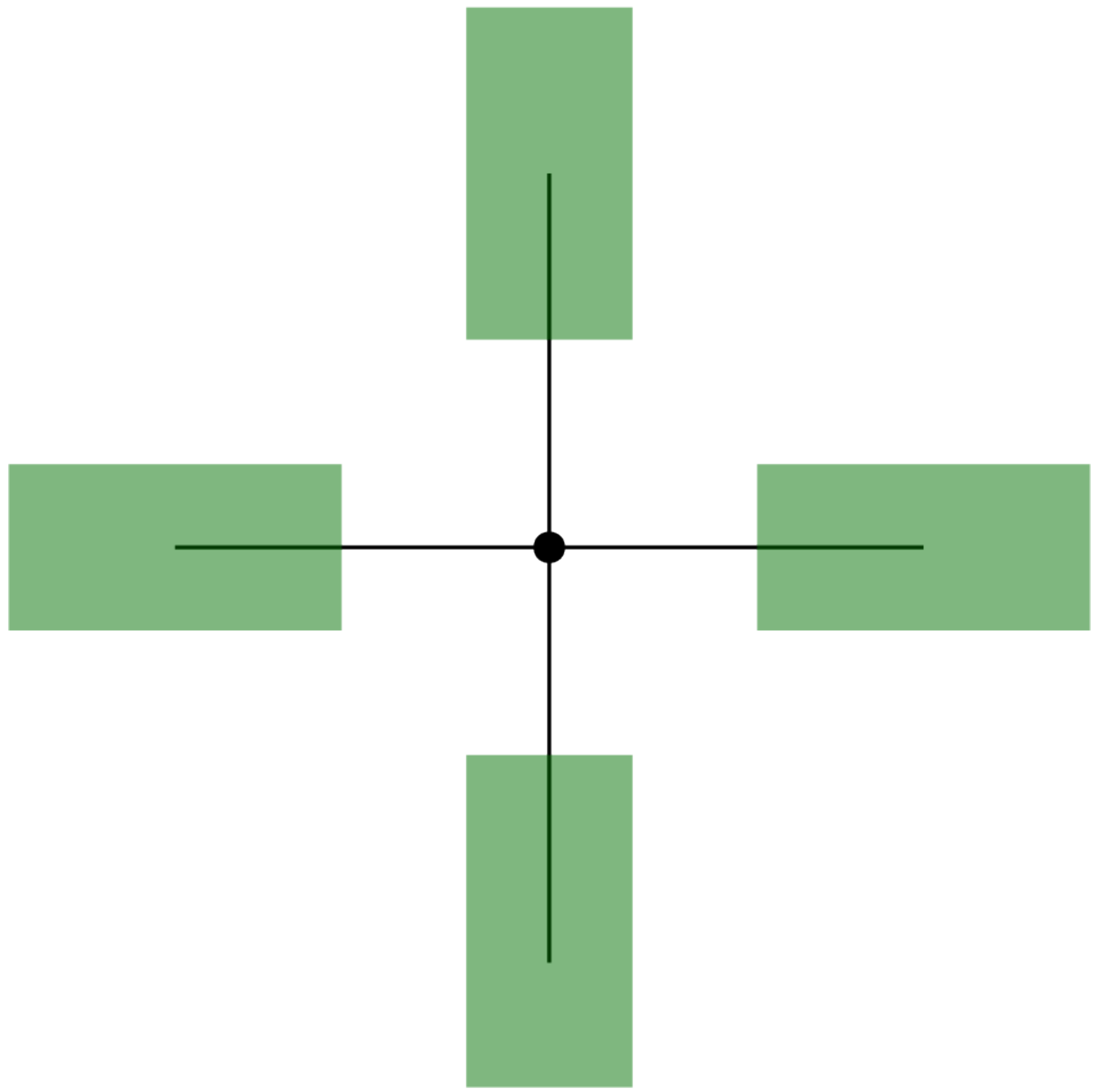
A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $$\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod 2 \text{ for all } v \in V$$

entering *and* exiting each vertex



Local Constraints of a Threading

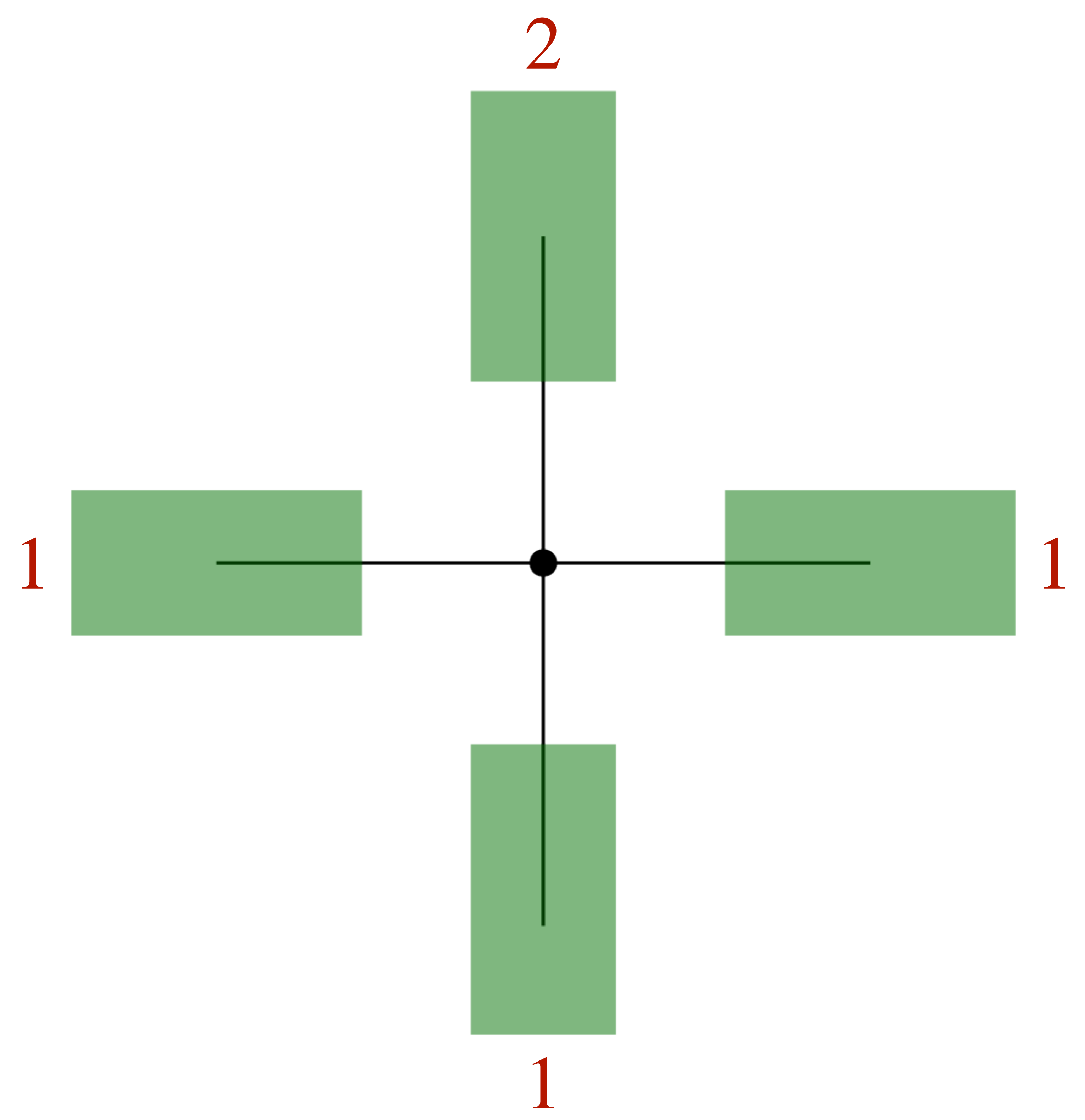
A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod 2$ for all $v \in V$

entering *and* exiting each vertex



$\sum_{u \in N(v)} x_{uv} = 5$ ✘

Local Constraints of a Threading

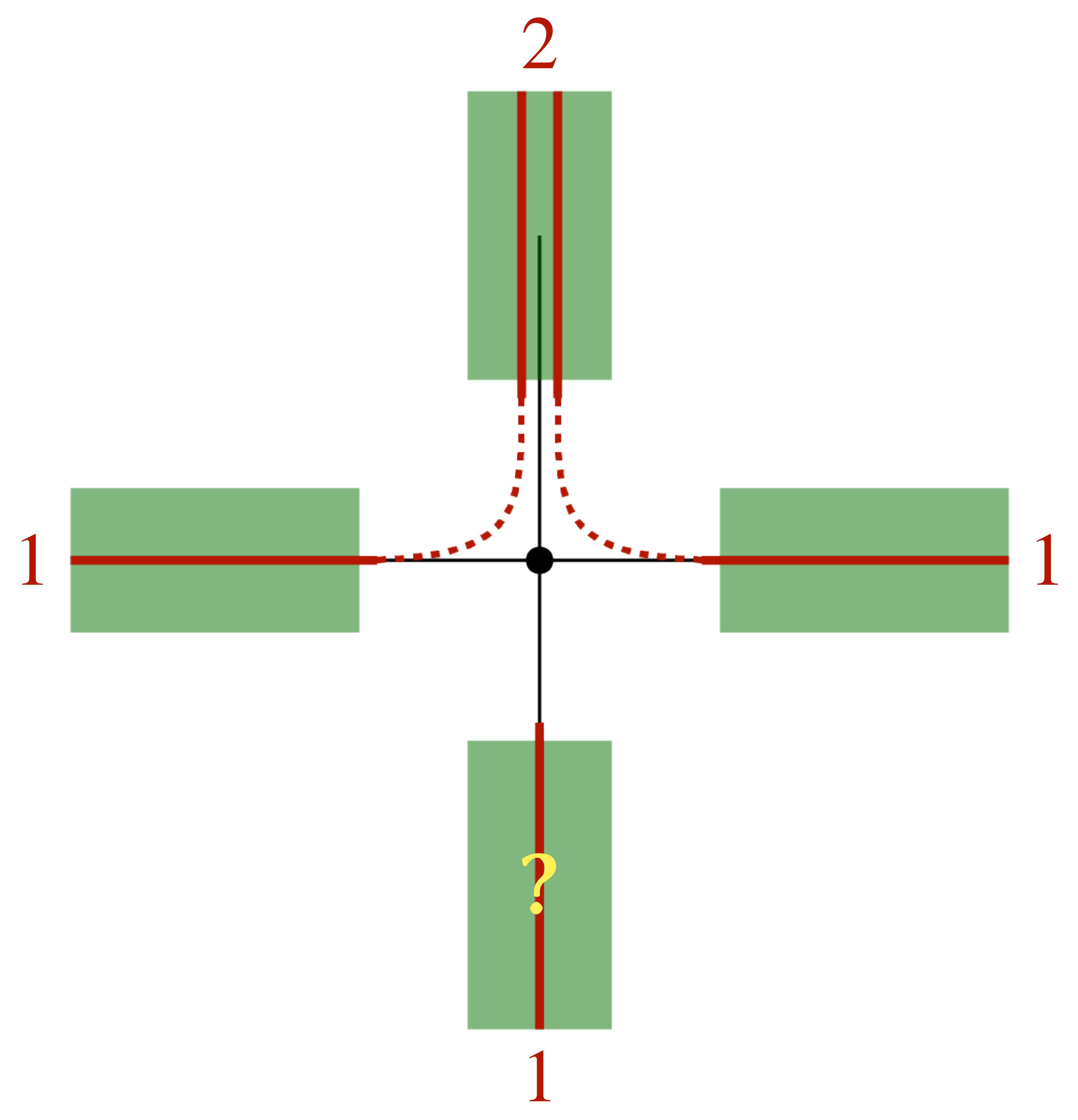
A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod 2$ for all $v \in V$

entering *and* exiting each vertex



$\sum_{u \in N(v)} x_{uv} = 5$ ✘

Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $\displaystyle\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod{2}$ for all $v \in V$

entering *and* exiting each vertex

(C3) $\displaystyle\sum_{w \in N(v) \setminus \{u\}} x_{wv} \geq x_{uv}$ for all $uv \in E$

without U-turning,

Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

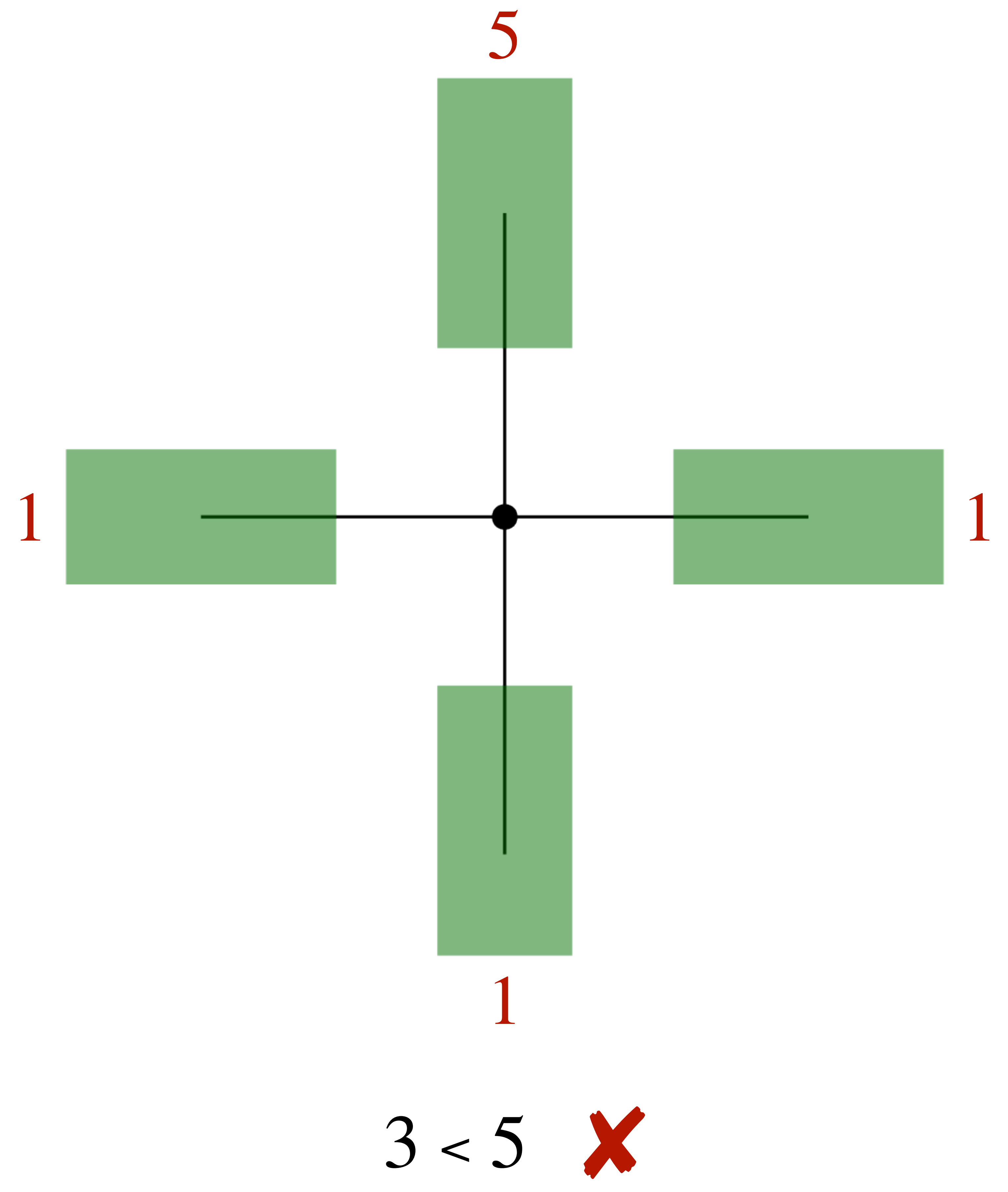
visits each edge at least once,

(C2) $\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod 2$ for all $v \in V$

entering *and* exiting each vertex

(C3) $\sum_{w \in N(v) \setminus \{u\}} x_{wv} \geq x_{uv}$ for all $uv \in E$

without U-turning,



Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

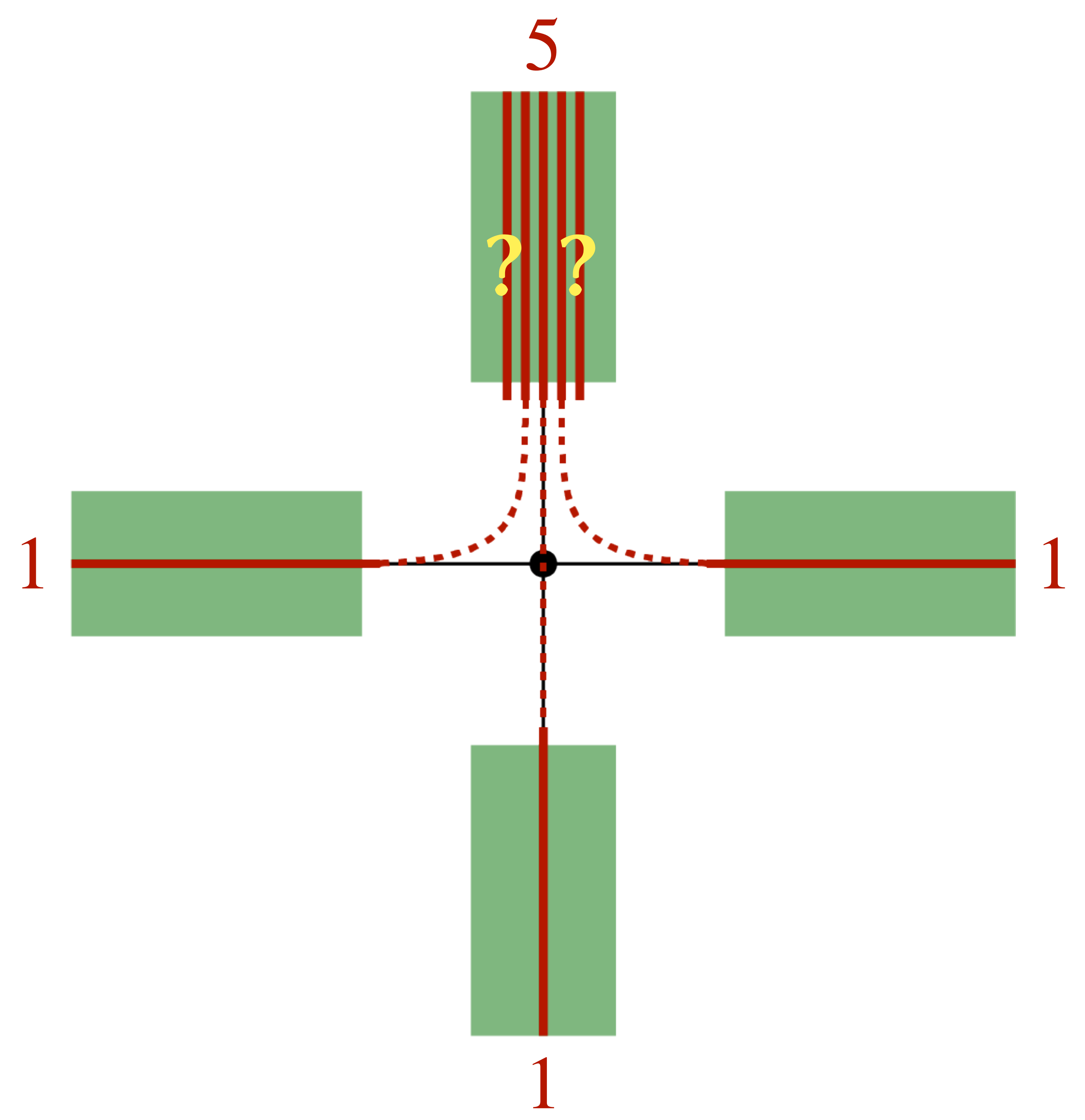
visits each edge at least once,

(C2) $\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod{2}$ for all $v \in V$

entering *and* exiting each vertex

(C3) $\sum_{w \in N(v) \setminus \{u\}} x_{wv} \geq x_{uv}$ for all $uv \in E$

without U-turning,



$3 < 5$ ✘

Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod 2$ for all $v \in V$

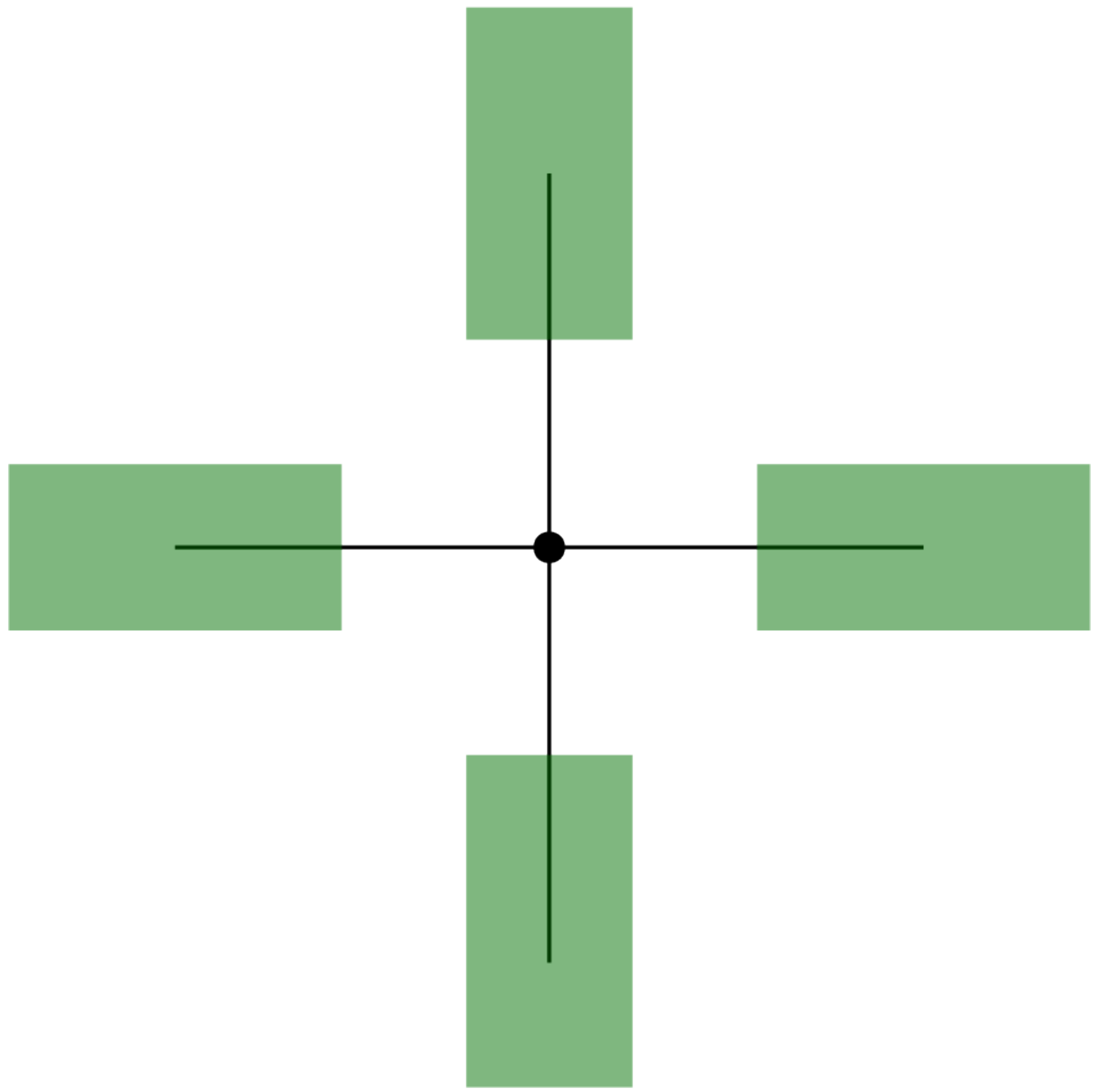
entering *and* exiting each vertex

(C3) $\sum_{w \in N(v) \setminus \{u\}} x_{wv} \geq x_{uv}$ for all $uv \in E$

without U-turning,

(C4) $\sum_{u \in N(v)} x_{uv} \geq 2(d(v) - 1)$ for all $v \in V$

enough times to form a spanning tree.



Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod 2$ for all $v \in V$

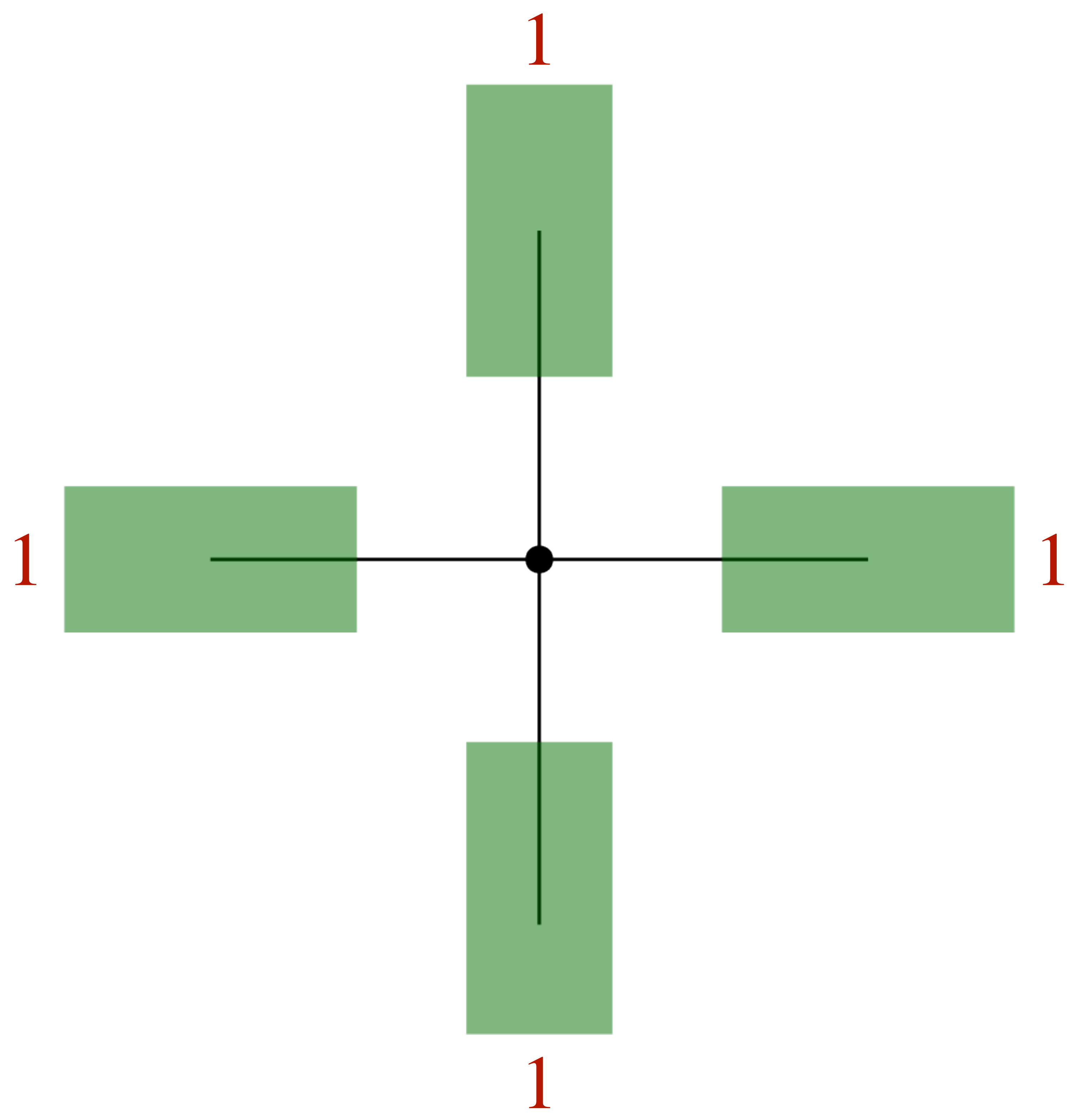
entering *and* exiting each vertex

(C3) $\sum_{w \in N(v) \setminus \{u\}} x_{wv} \geq x_{uv}$ for all $uv \in E$

without U-turning,

(C4) $\sum_{u \in N(v)} x_{uv} \geq 2(d(v) - 1)$ for all $v \in V$

enough times to form a spanning tree.



$\sum_{u \in N(v)} x_{uv} = 4 < 6$ ✘

Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $$\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod 2 \text{ for all } v \in V$$

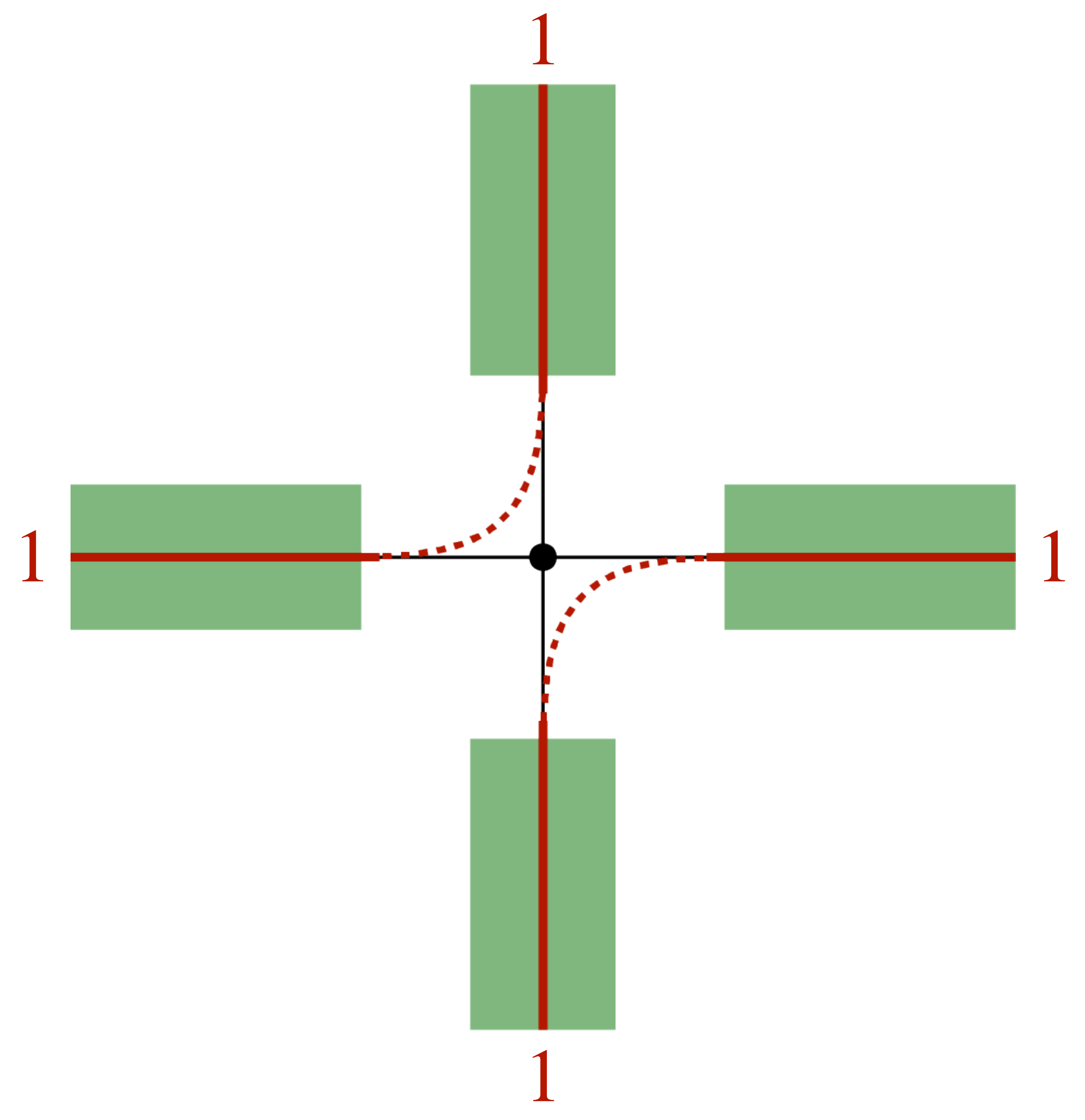
entering *and* exiting each vertex

(C3) $$\sum_{w \in N(v) \setminus \{u\}} x_{wv} \geq x_{uv} \text{ for all } uv \in E$$

without U-turning,

(C4) $$\sum_{u \in N(v)} x_{uv} \geq 2(d(v) - 1) \text{ for all } v \in V$$

enough times to form a spanning tree.



$$\sum_{u \in N(v)} x_{uv} = 4 < 6$$ ✘

Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod 2$ for all $v \in V$

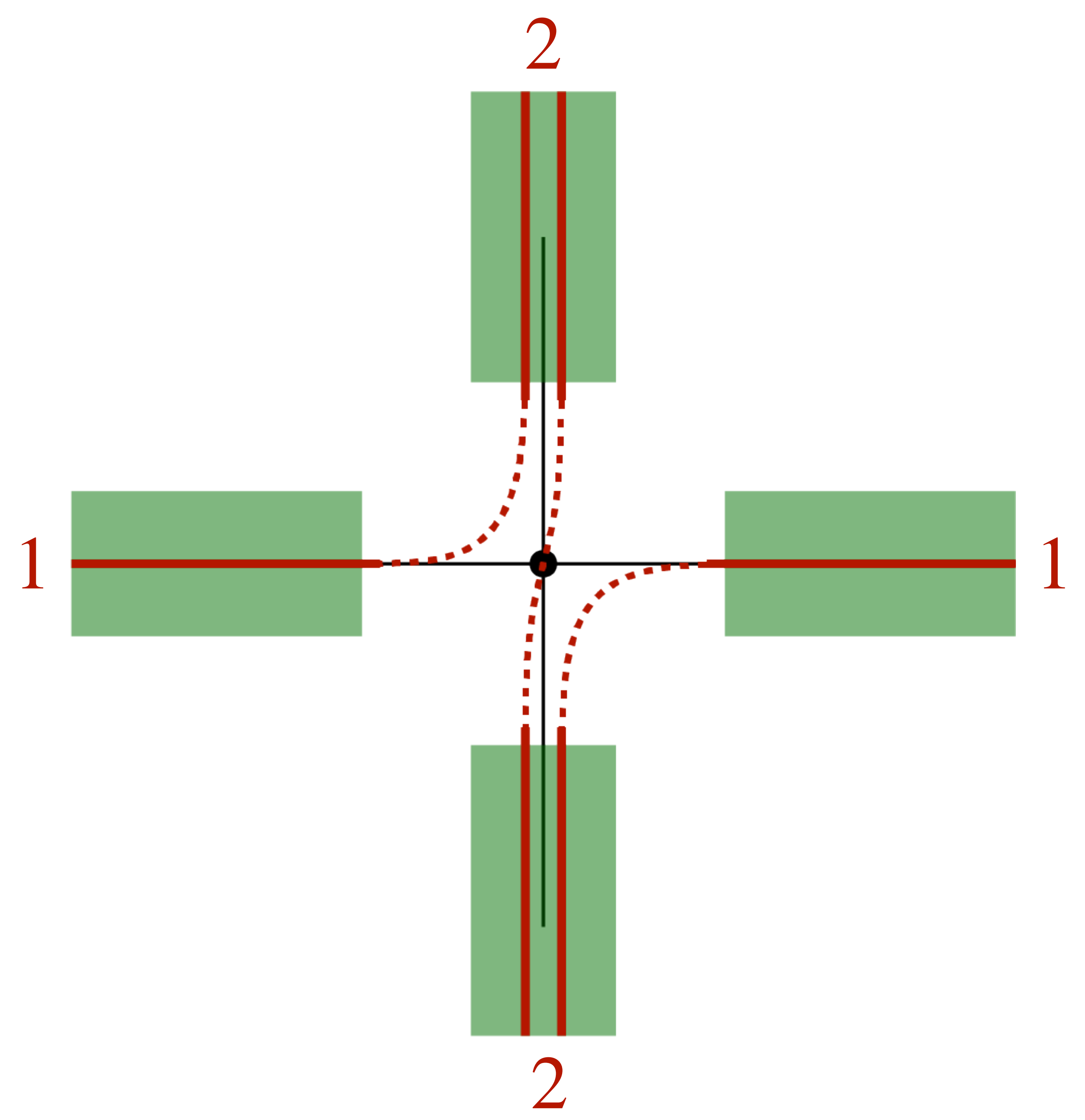
entering *and* exiting each vertex

(C3) $\sum_{w \in N(v) \setminus \{u\}} x_{wv} \geq x_{uv}$ for all $uv \in E$

without U-turning,

(C4) $\sum_{u \in N(v)} x_{uv} \geq 2(d(v) - 1)$ for all $v \in V$

enough times to form a spanning tree.



$\sum_{u \in N(v)} x_{uv} = 6$ ✓

Local Constraints of a Threading

A **local threading** of G consists of counts $\{x_{uv}\}$ satisfying constraints

(C1) $x_{uv} \geq 1$ for all $uv \in E$

visits each edge at least once,

(C2) $\displaystyle\sum_{u \in N(v)} x_{uv} \equiv 0 \pmod{2}$ for all $v \in V$

entering *and* exiting each vertex

(C3) $\displaystyle\sum_{w \in N(v) \setminus \{u\}} x_{wv} \geq x_{uv}$ for all $uv \in E$

without U-turning,

(C4) $\displaystyle\sum_{u \in N(v)} x_{uv} \geq 2(d(v) - 1)$ for all $v \in V$

enough times to form a spanning tree.

Theorem. We can construct a threading T of G from a local threading $\{x_{uv}\}$ of G such that T visits edge uv exactly x_{uv} times. Hence $|T| = \sum_{uv \in E} x_{uv}$.

Theorem. We can construct a threading T of G from a local threading $\{x_{uv}\}$ of G such that T visits edge uv exactly x_{uv} times. Hence $|T| = \sum_{uv \in E} x_{uv}$.

Game Plan. Constructive proof in two steps:

Theorem. We can construct a threading T of G from a local threading $\{x_{uv}\}$ of G such that T visits edge uv exactly x_{uv} times. Hence $|T| = \sum_{uv \in E} x_{uv}$.

Game Plan. Constructive proof in two steps:

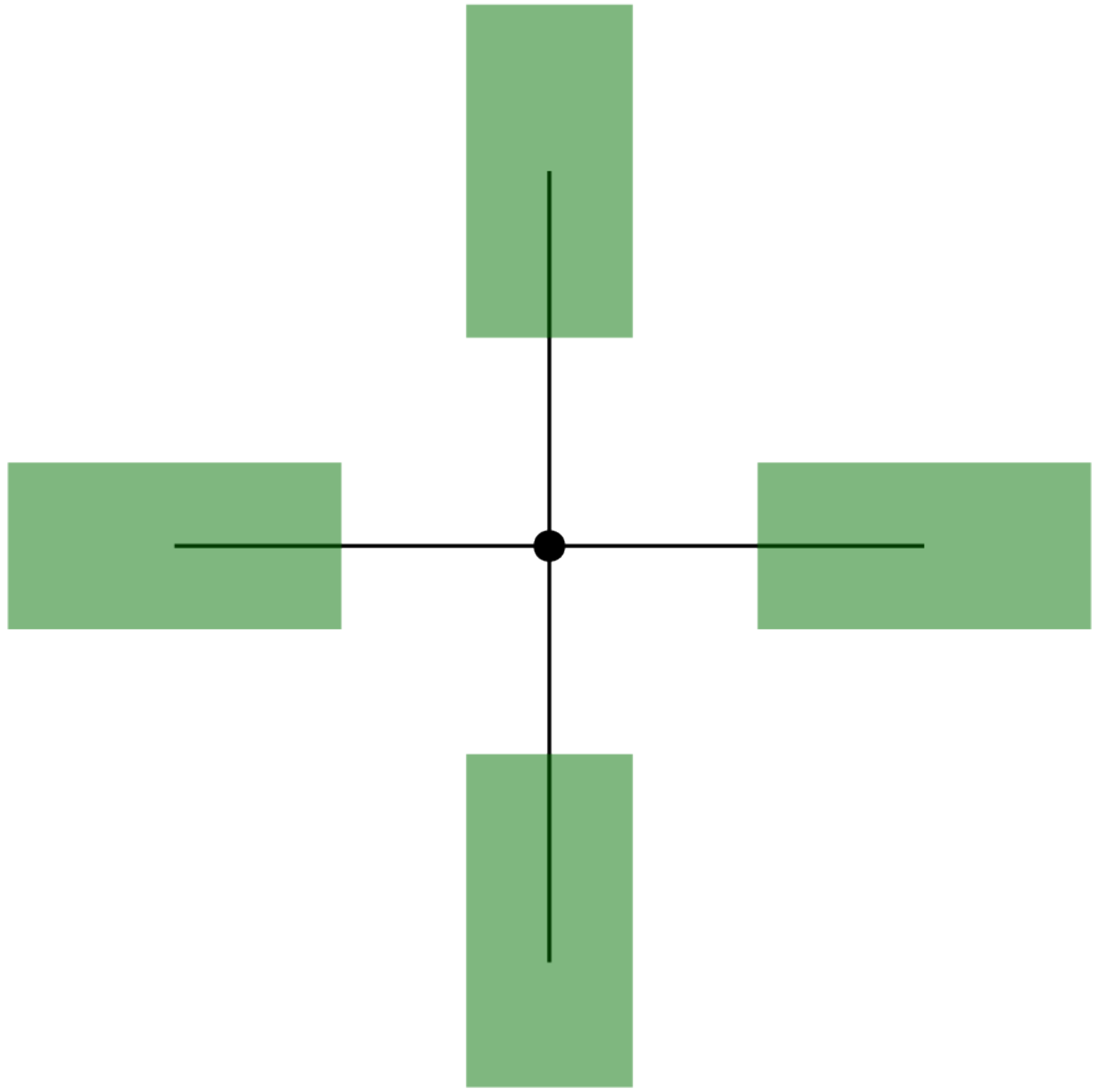
1. Compute a connected junction graph at every vertex given a local threading

Theorem. We can construct a threading T of G from a local threading $\{x_{uv}\}$ of G such that T visits edge uv exactly x_{uv} times. Hence $|T| = \sum_{uv \in E} x_{uv}$.

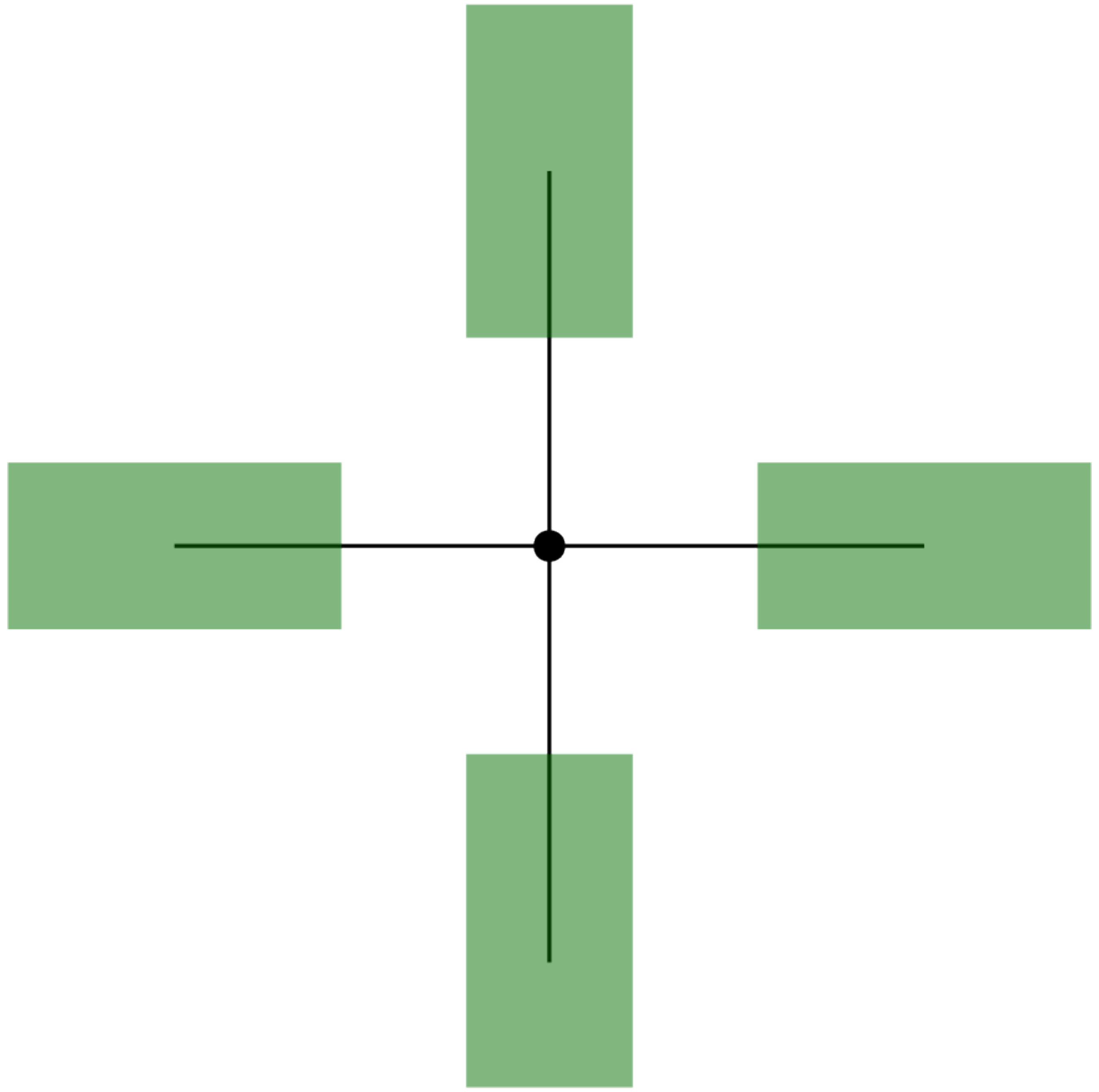
Game Plan. Constructive proof in two steps:

1. Compute a connected junction graph at every vertex given a local threading
2. Find a threading from the resulting collection of junction graphs

Step 1: Construct a Connected Junction Graph

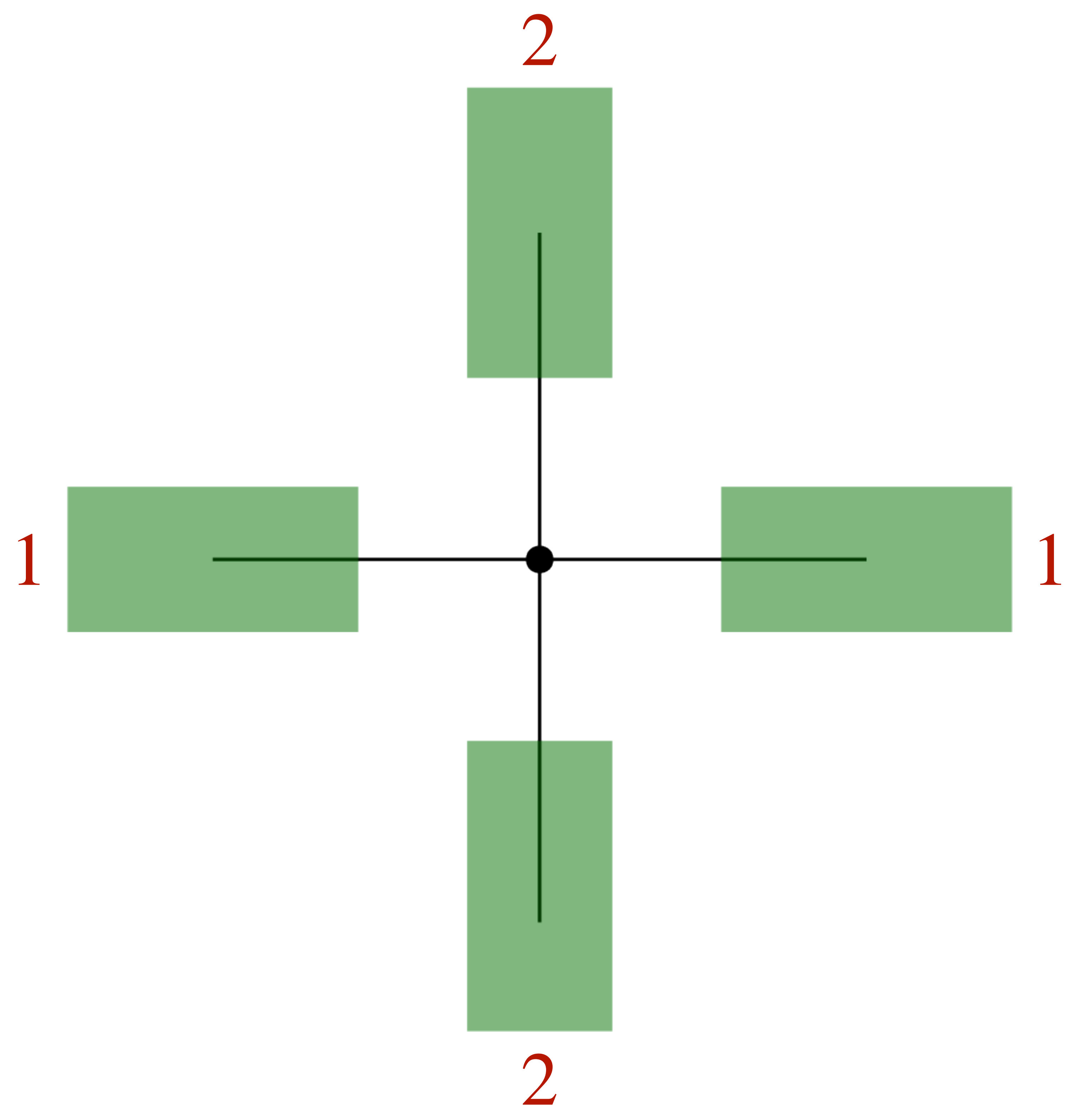


Step 1: Construct a Connected Junction Graph



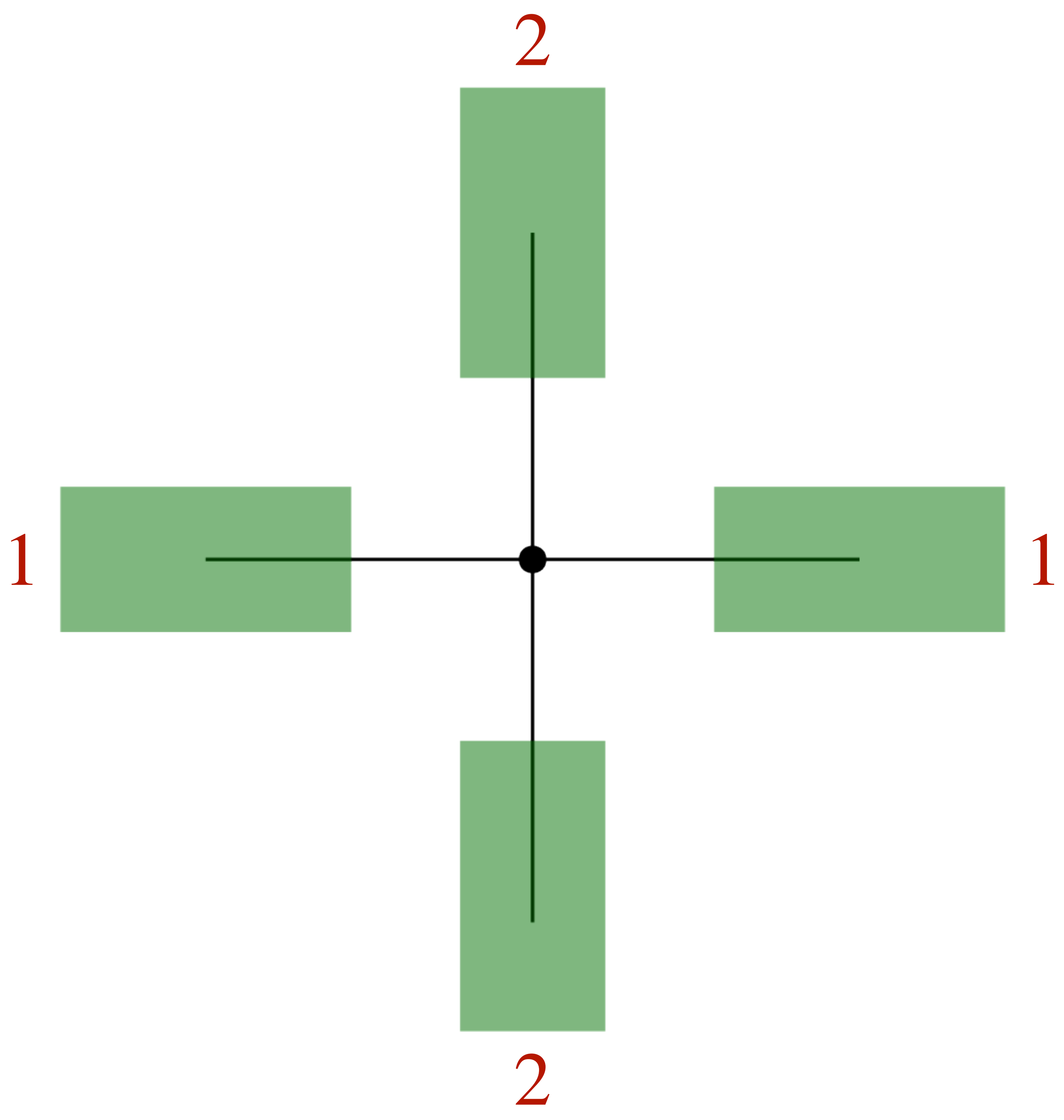
Given a local threading $\{x_{uv}\}$:

Step 1: Construct a Connected Junction Graph



Given a local threading $\{x_{uv}\}$:

Step 1: Construct a Connected Junction Graph

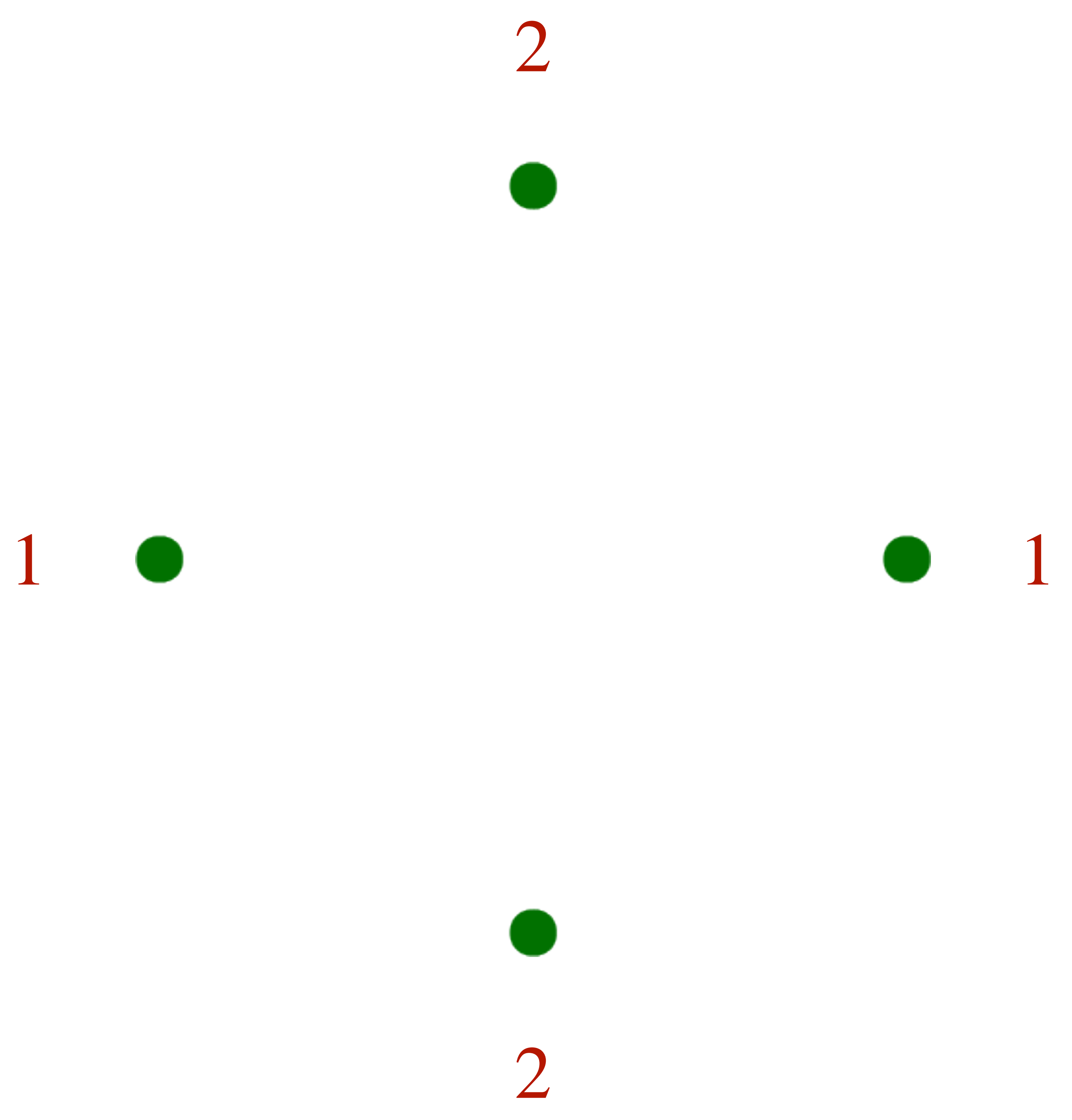


Given a local threading $\{x_{uv}\}$:

Consider the case where $\sum_{u \in N(v)} x_{uv} =^{*} 2(d(v)-1)$ …

* Refer to paper for generalization to $\geq$

Step 1: Construct a Connected Junction Graph

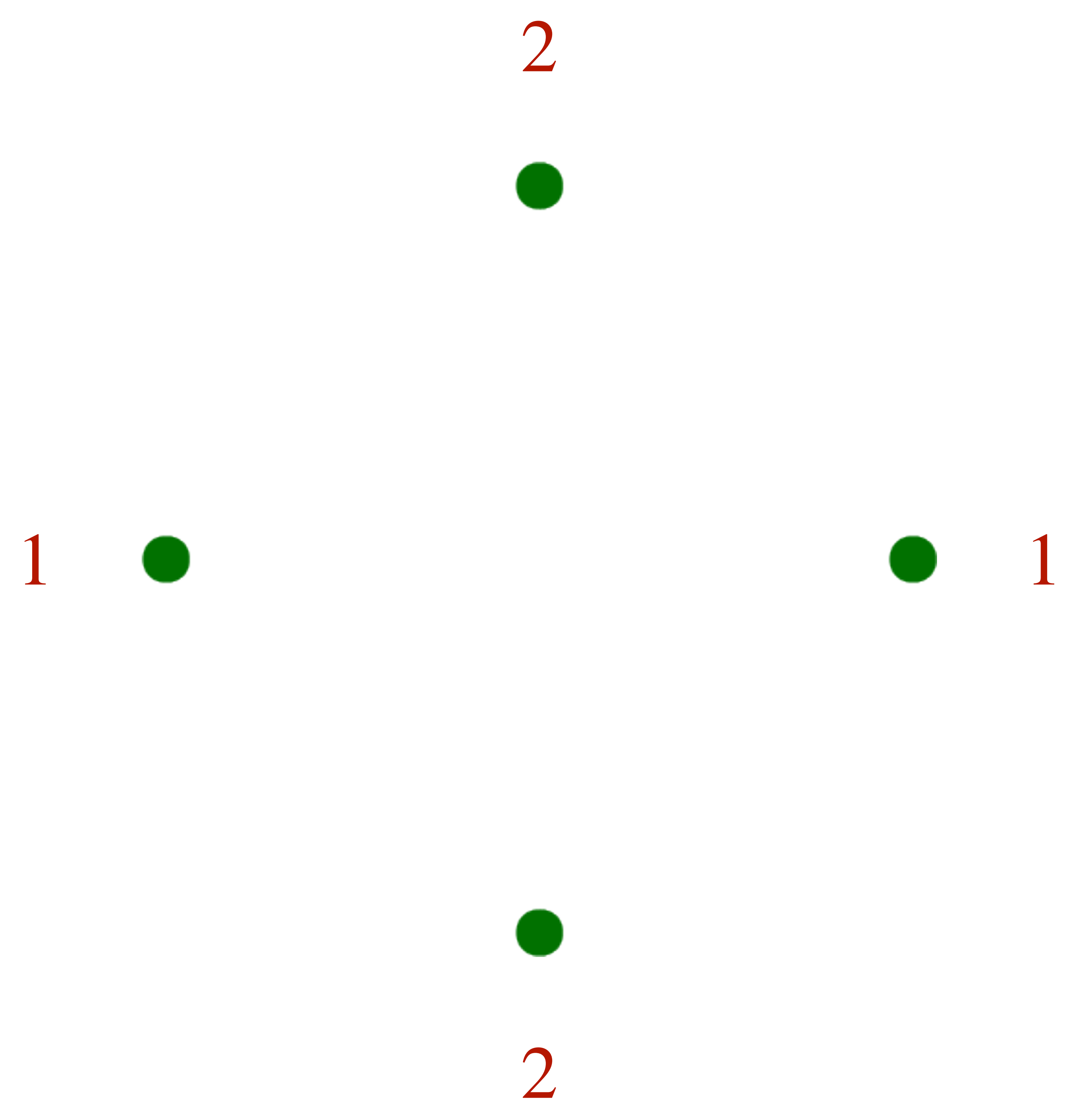


Given a local threading $\{x_{uv}\}$:

Consider the case where $\sum_{u\in N(v)} x_{uv} = 2(d(v)-1)$ ...

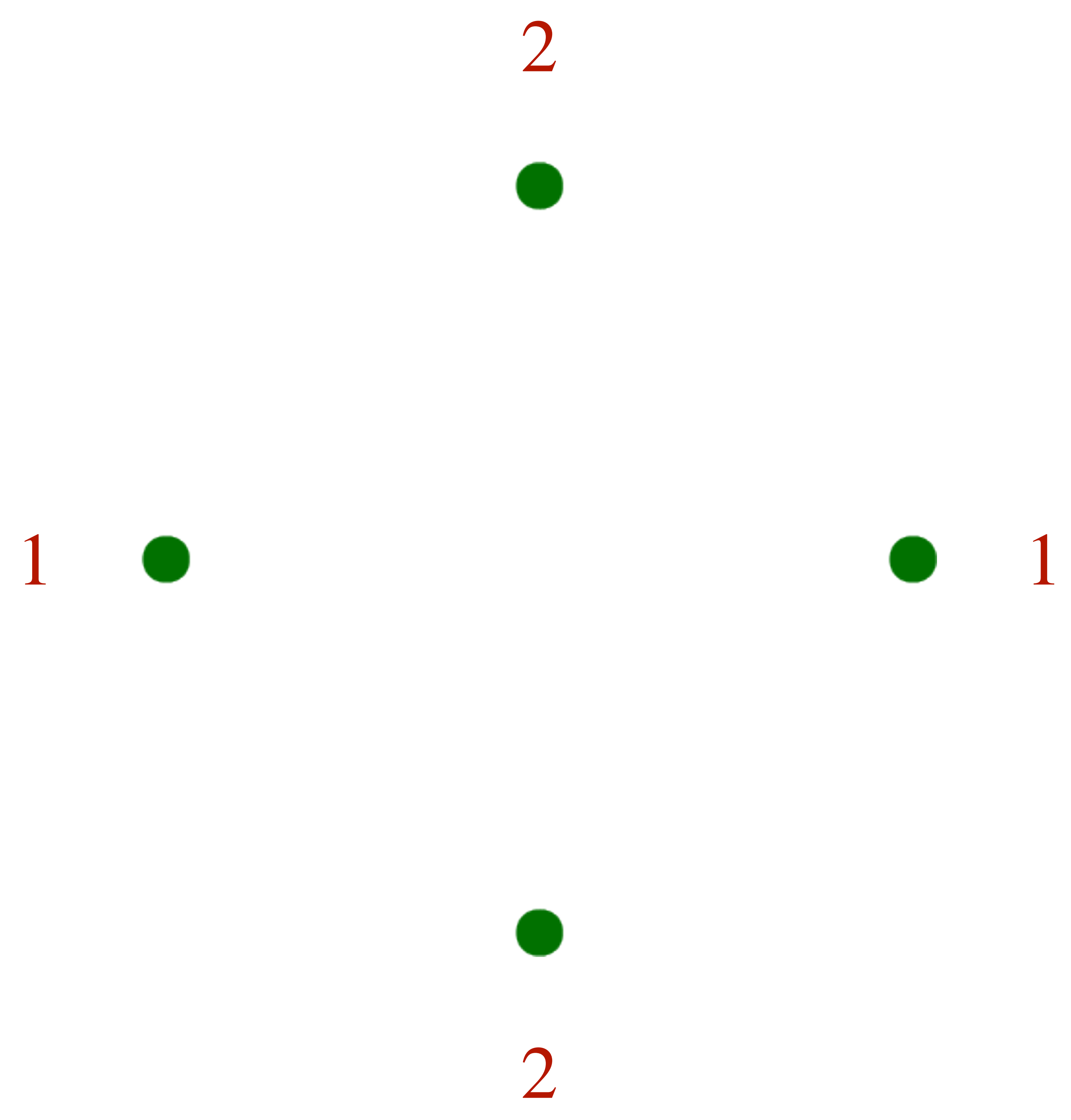
Step 1: Construct a Connected Junction Graph

Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.



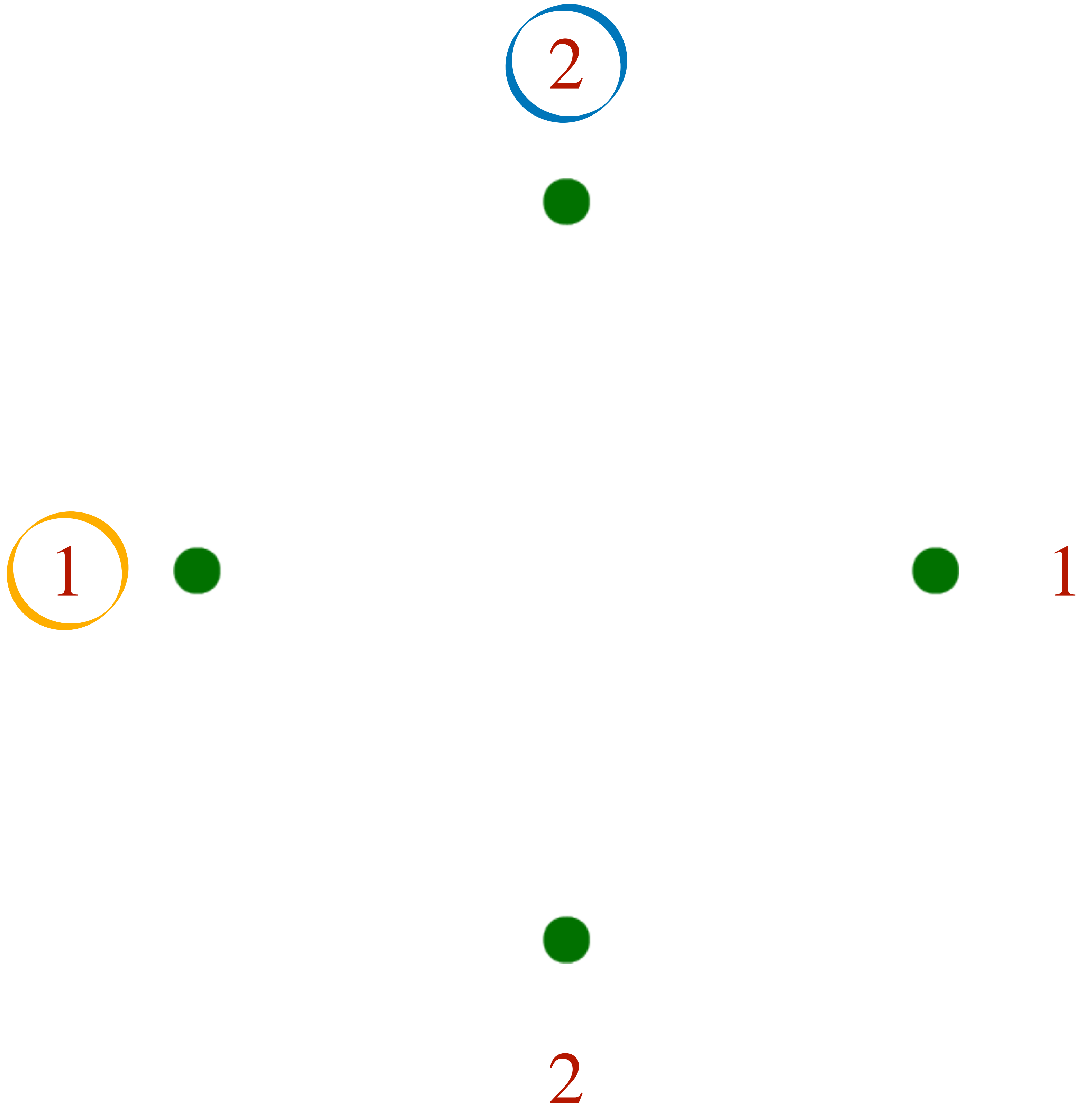
Step 1: Construct a Connected Junction Graph

Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.



Recursion:

Step 1: Construct a Connected Junction Graph

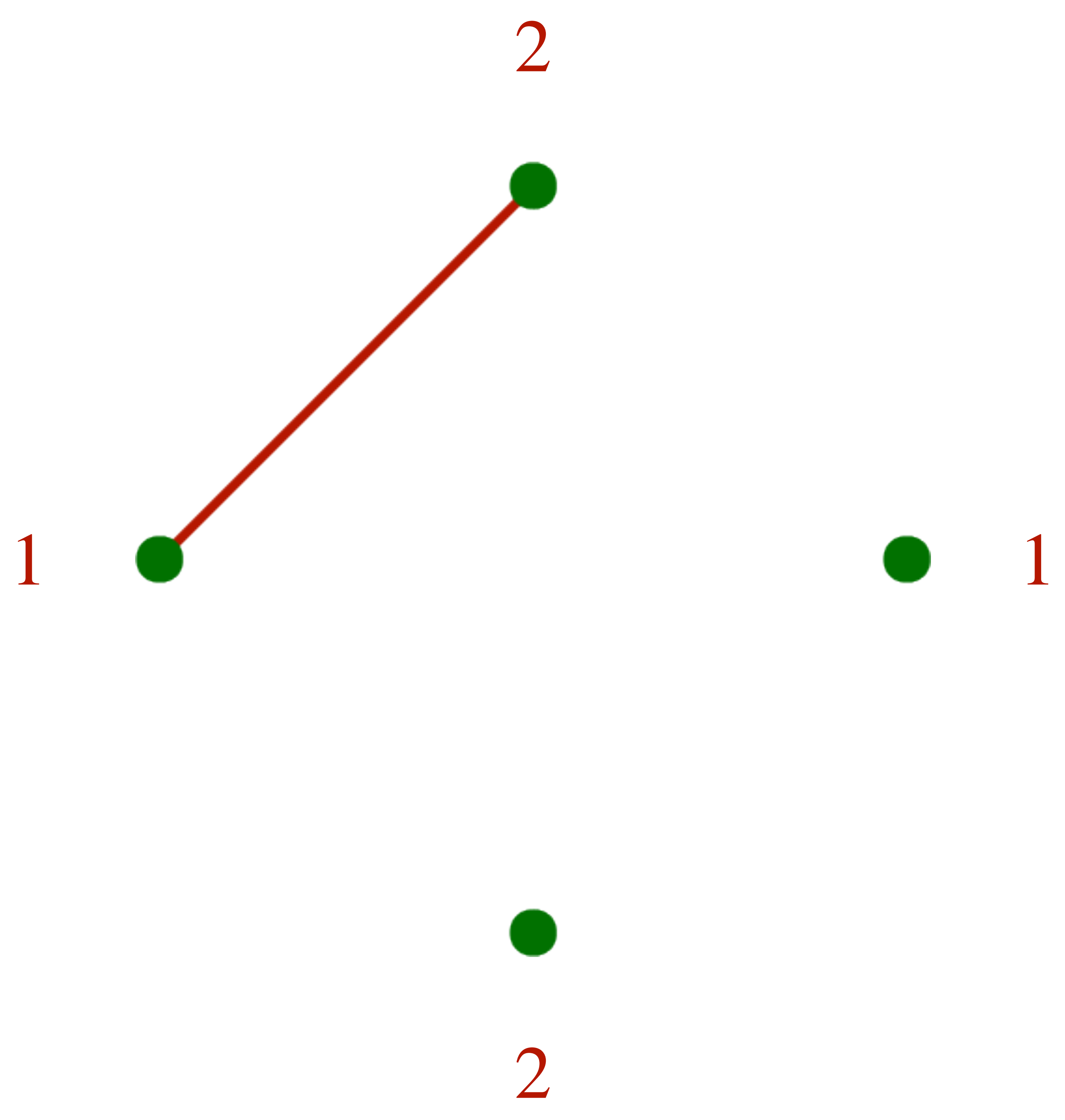


Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.

Recursion:

- Choose $x_i > 1$ and $x_j = 1$

Step 1: Construct a Connected Junction Graph

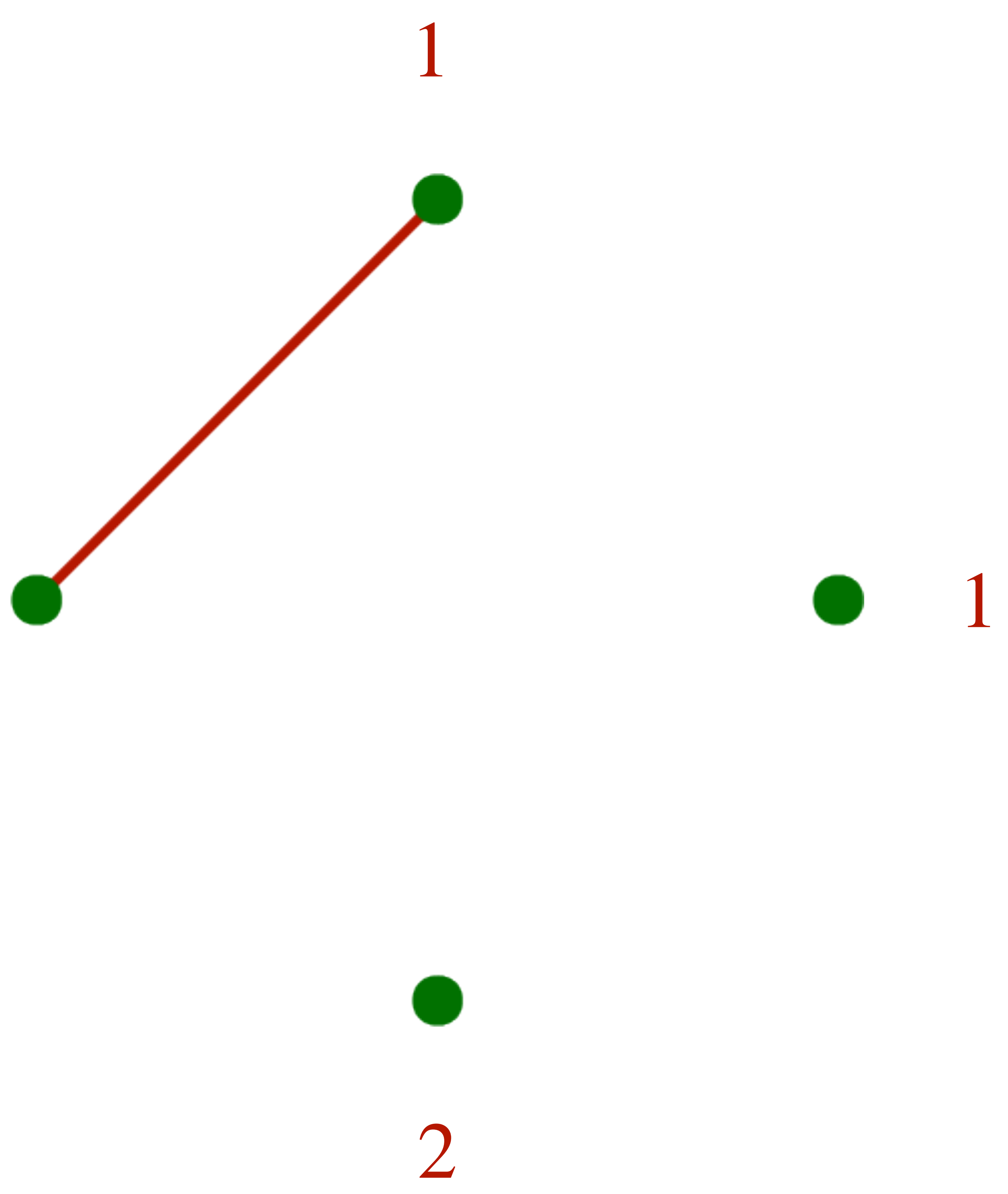


Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.

Recursion:

- Choose $x_i > 1$ and $x_j = 1$
- Connect vertices i and j

Step 1: Construct a Connected Junction Graph



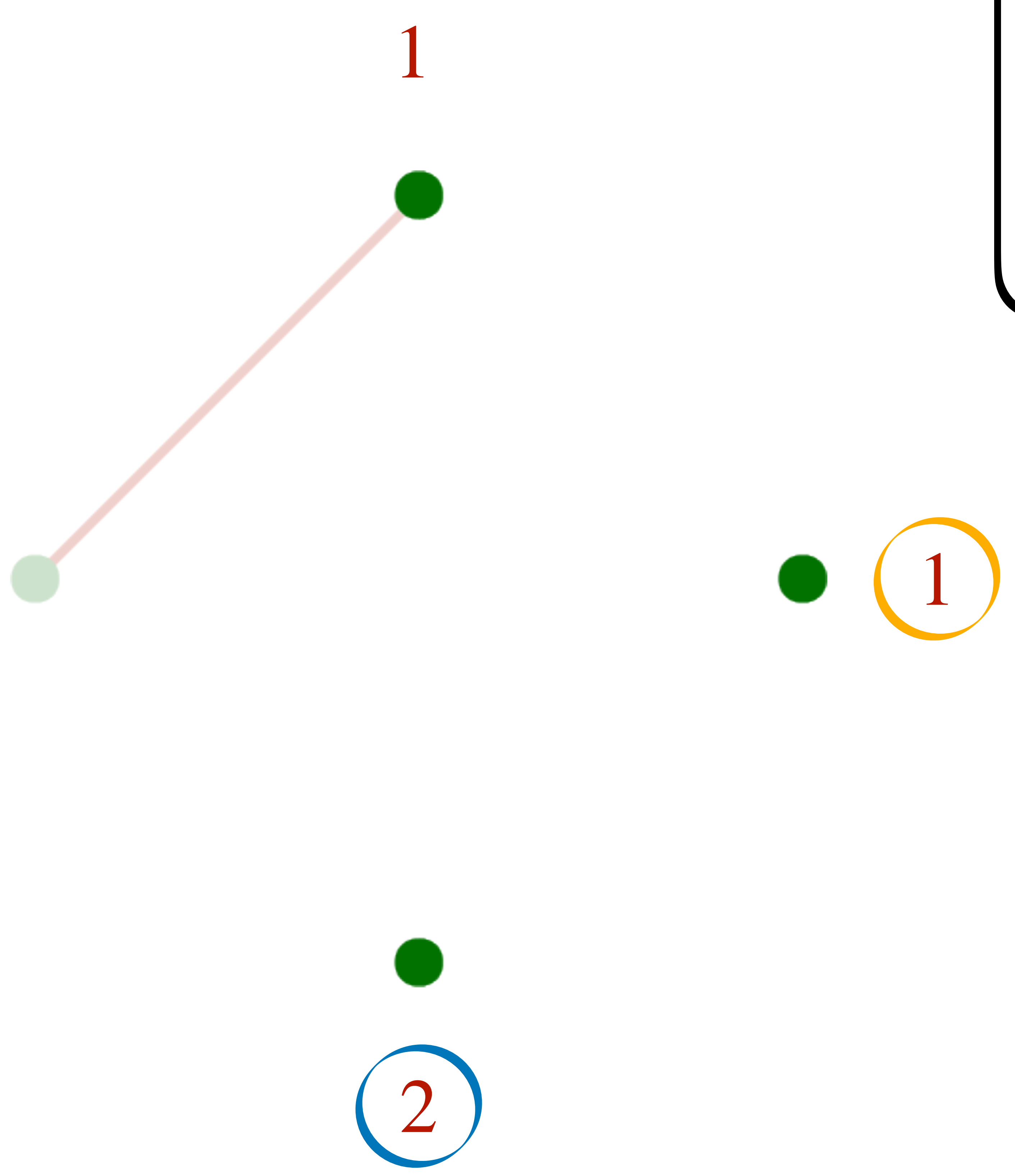
Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.

Recursion:

- Choose $x_i > 1$ and $x_j = 1$
- Connect vertices i and j
- Set $x_i = x_i - 1$, and x_j no longer exists

Step 1: Construct a Connected Junction Graph

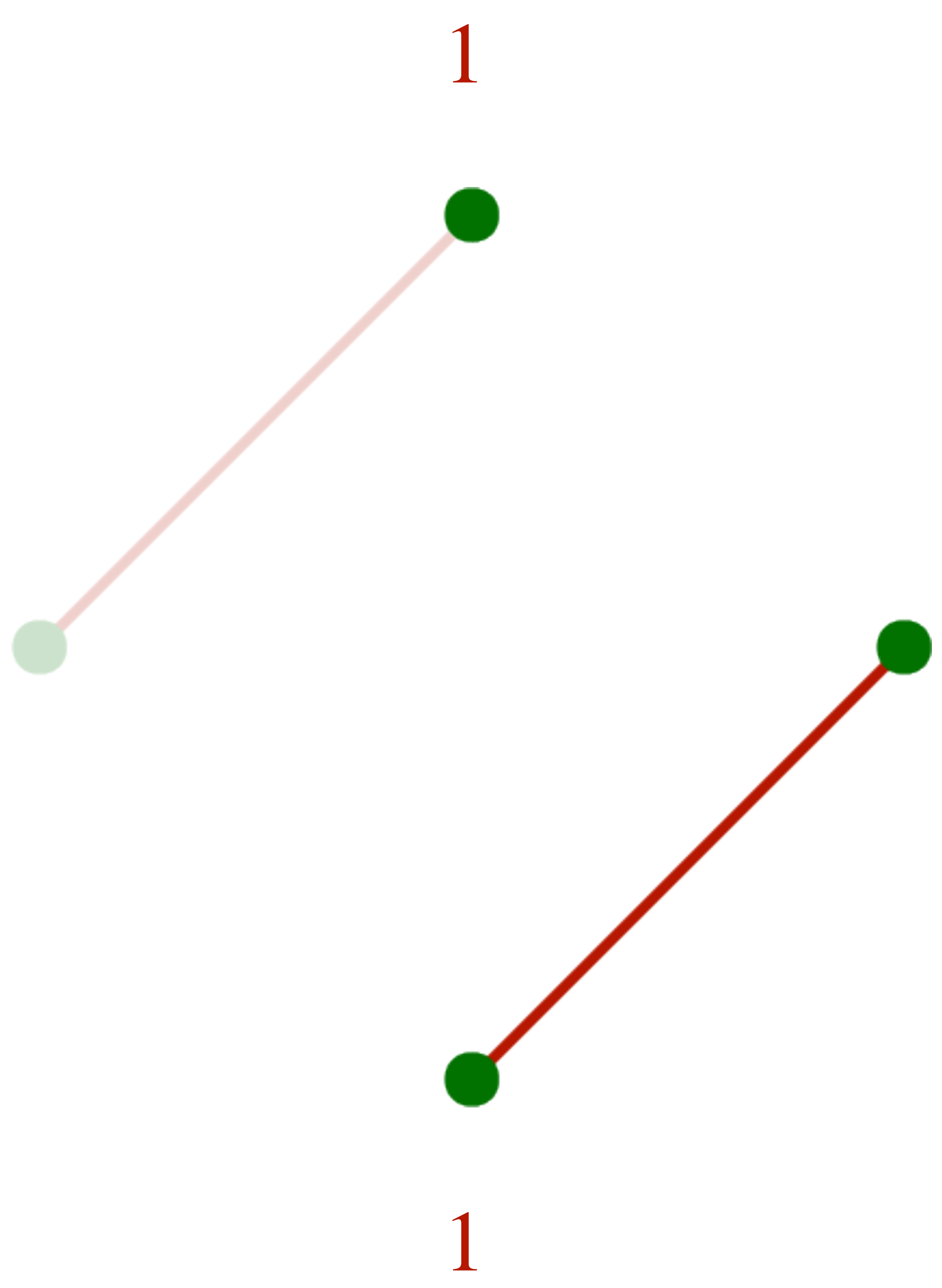
Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.



Recursion:

- Choose $x_i > 1$ and $x_j = 1$
- Connect vertices i and j
- Set $x_i = x_i - 1$, and x_j no longer exists

Step 1: Construct a Connected Junction Graph



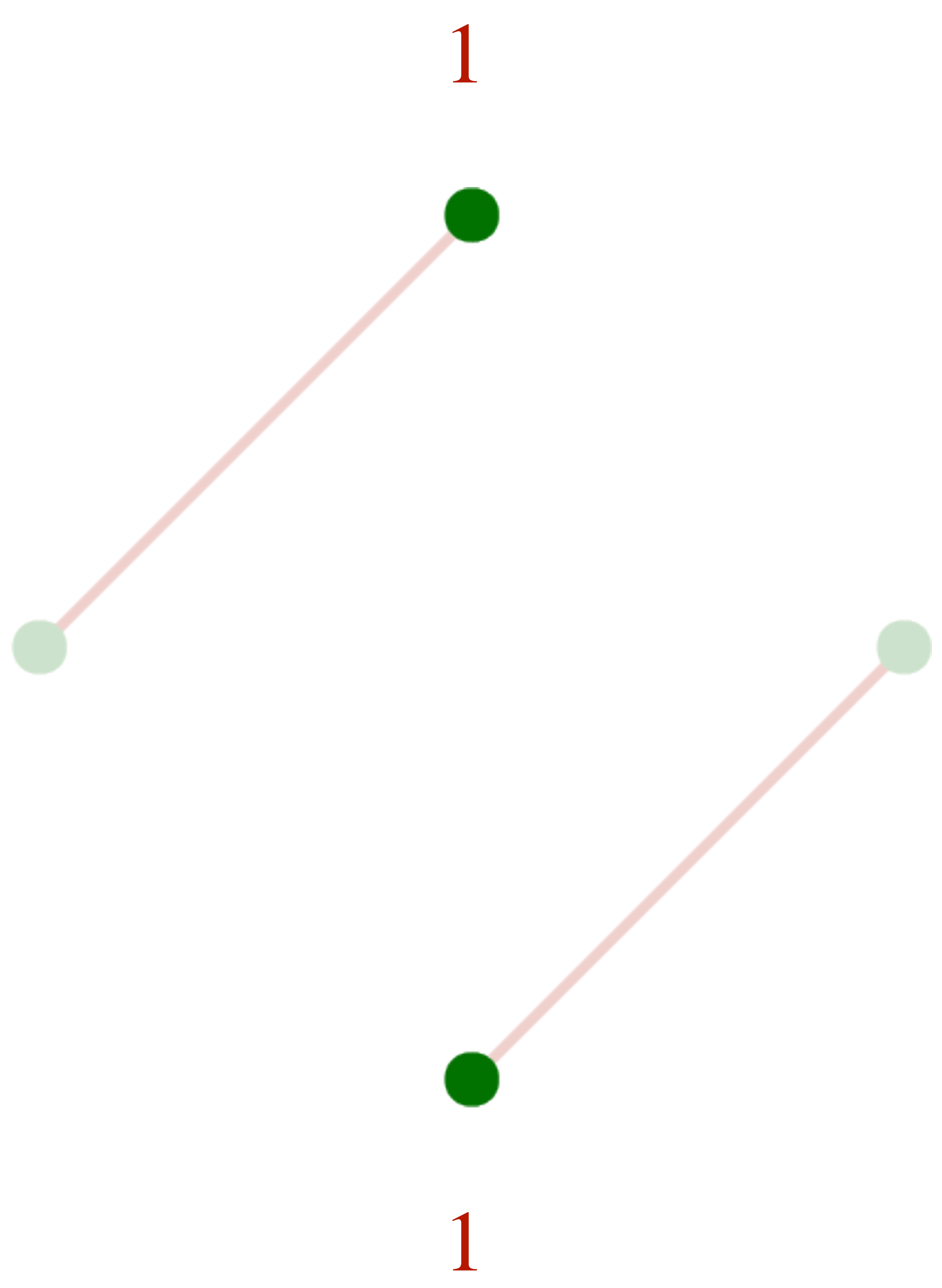
Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.

Recursion:

- Choose $x_i > 1$ and $x_j = 1$
- Connect vertices i and j
- Set $x_i = x_i - 1$, and x_j no longer exists

Step 1: Construct a Connected Junction Graph

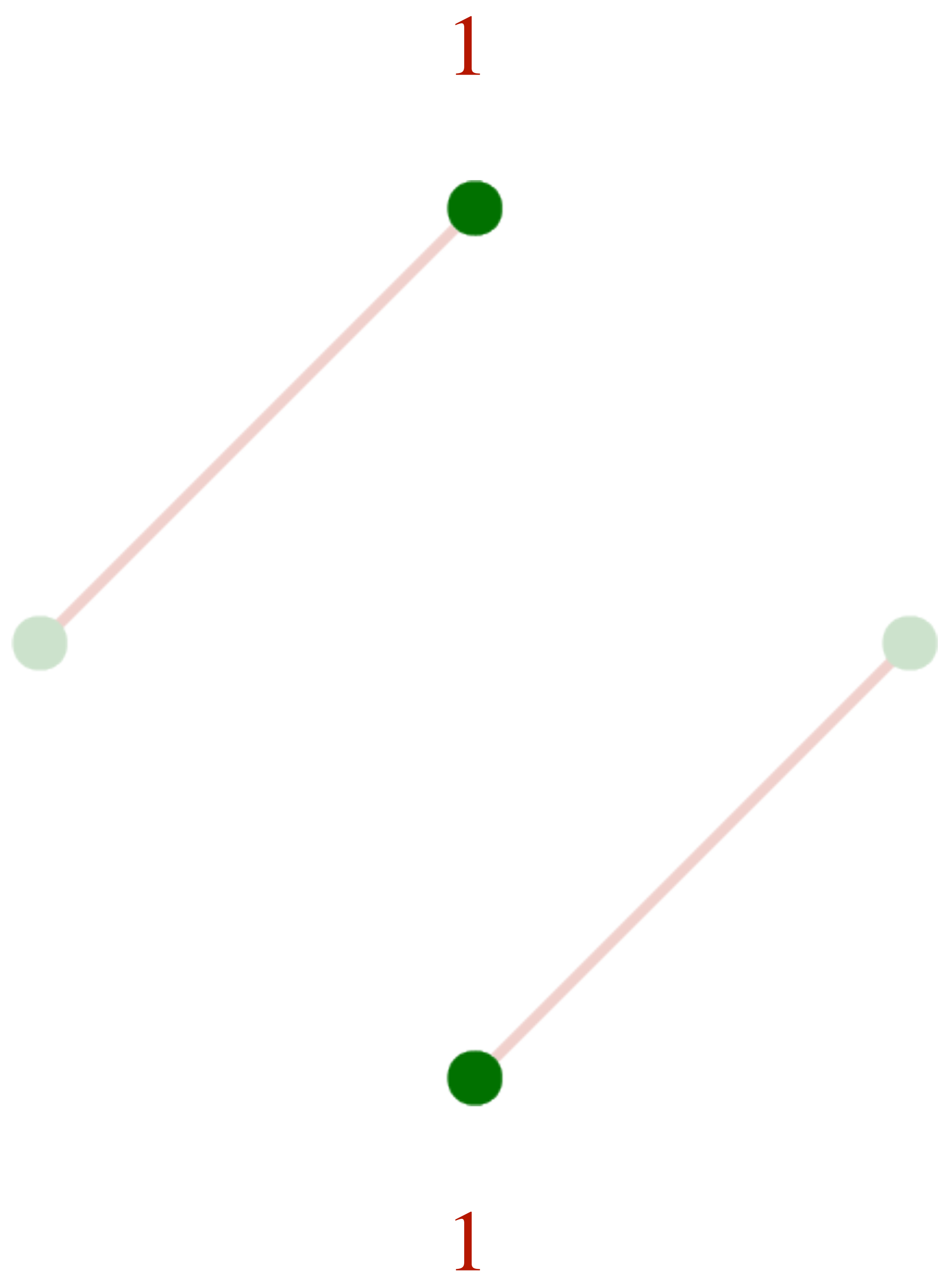
Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.



Recursion:

- Choose $x_i > 1$ and $x_j = 1$
- Connect vertices i and j
- Set $x_i = x_i - 1$, and x_j no longer exists

Step 1: Construct a Connected Junction Graph



Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.

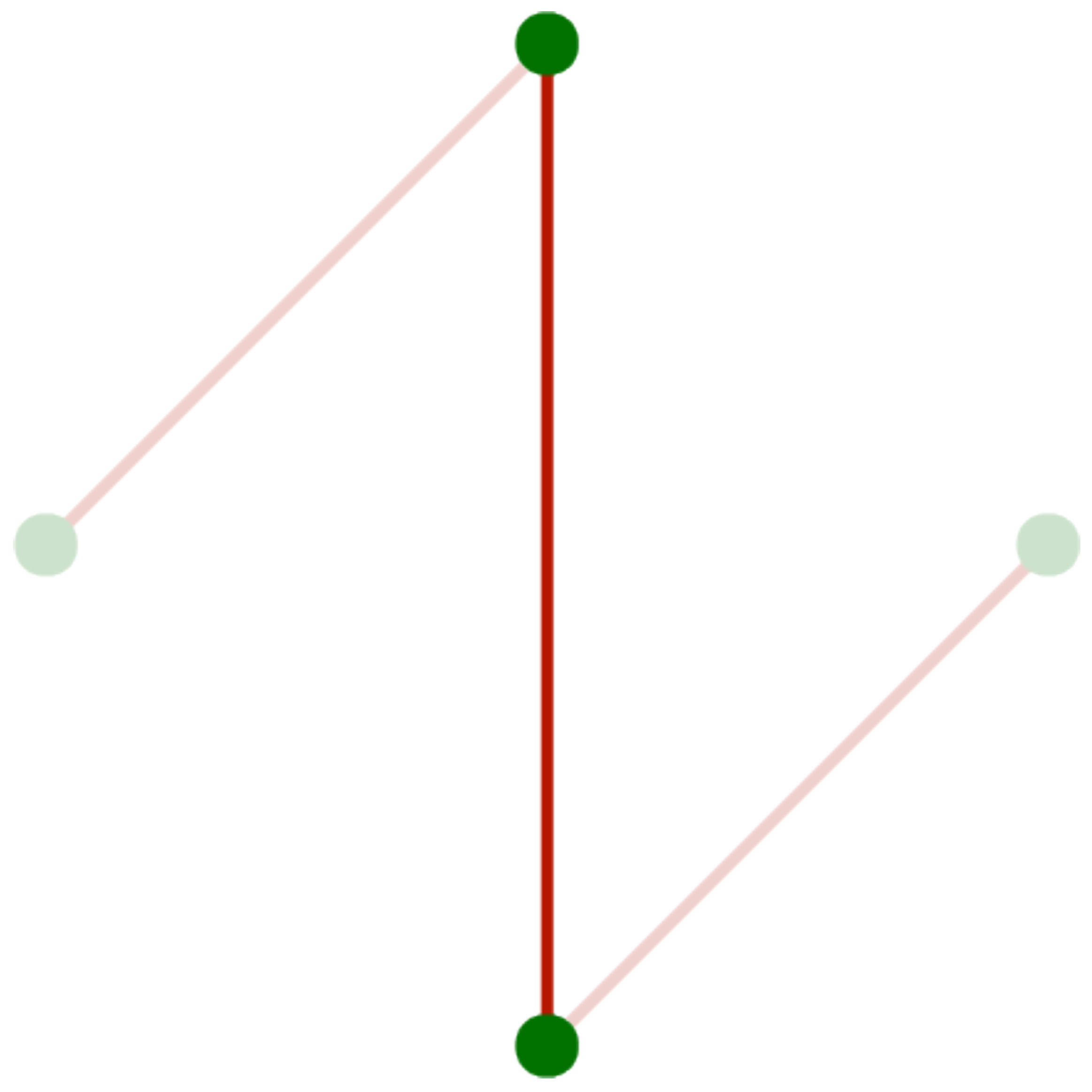
Recursion:

- Choose $x_i > 1$ and $x_j = 1$
- Connect vertices i and j
- Set $x_i = x_i - 1$, and x_j no longer exists

Base Case: Create a one-edge path since $d = 2$

Step 1: Construct a Connected Junction Graph

Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.

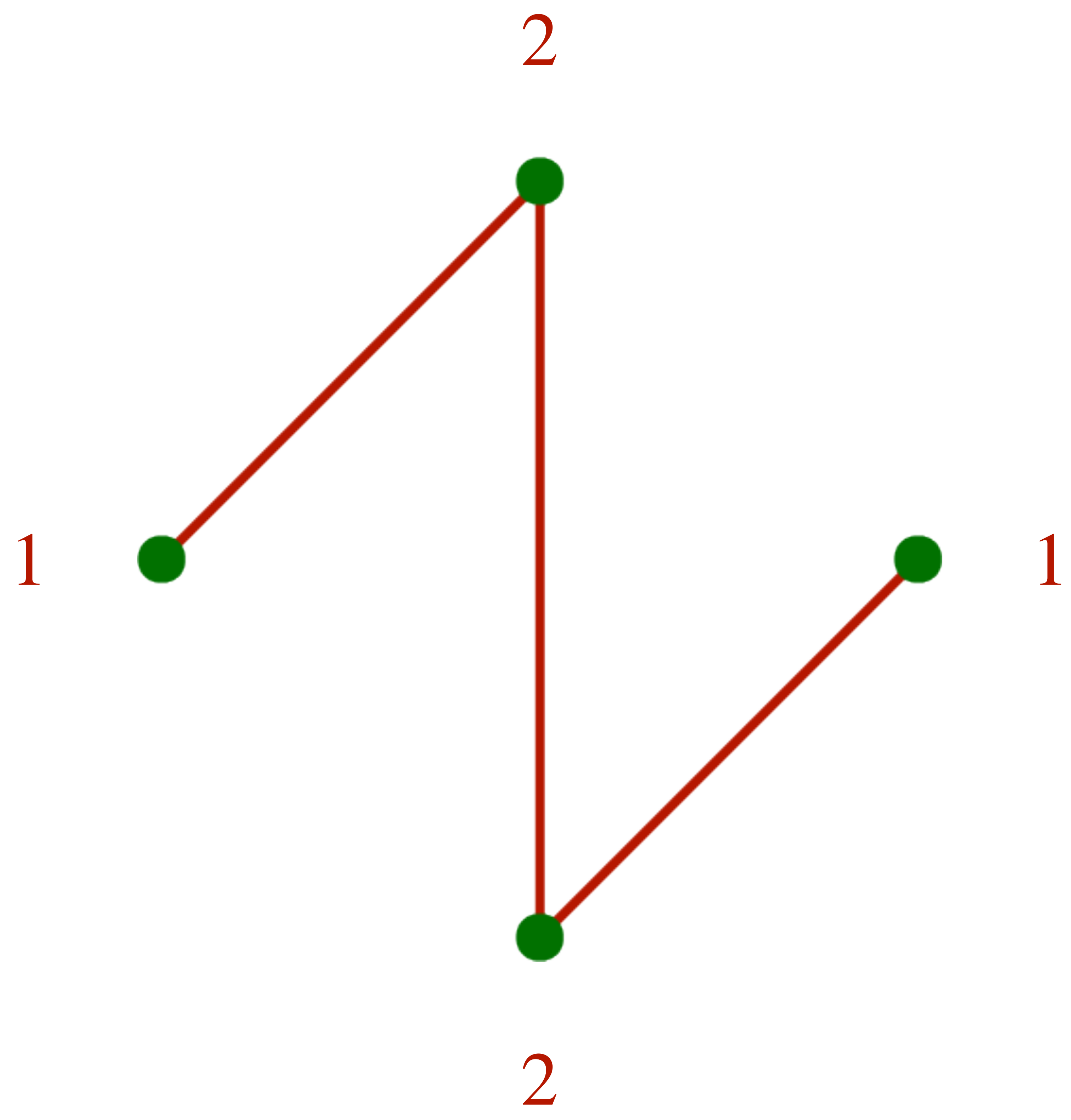


Recursion:

- Choose $x_i > 1$ and $x_j = 1$
- Connect vertices i and j
- Set $x_i = x_i - 1$, and x_j no longer exists

Base Case: Create a one-edge path since $d = 2$

Step 1: Construct a Connected Junction Graph

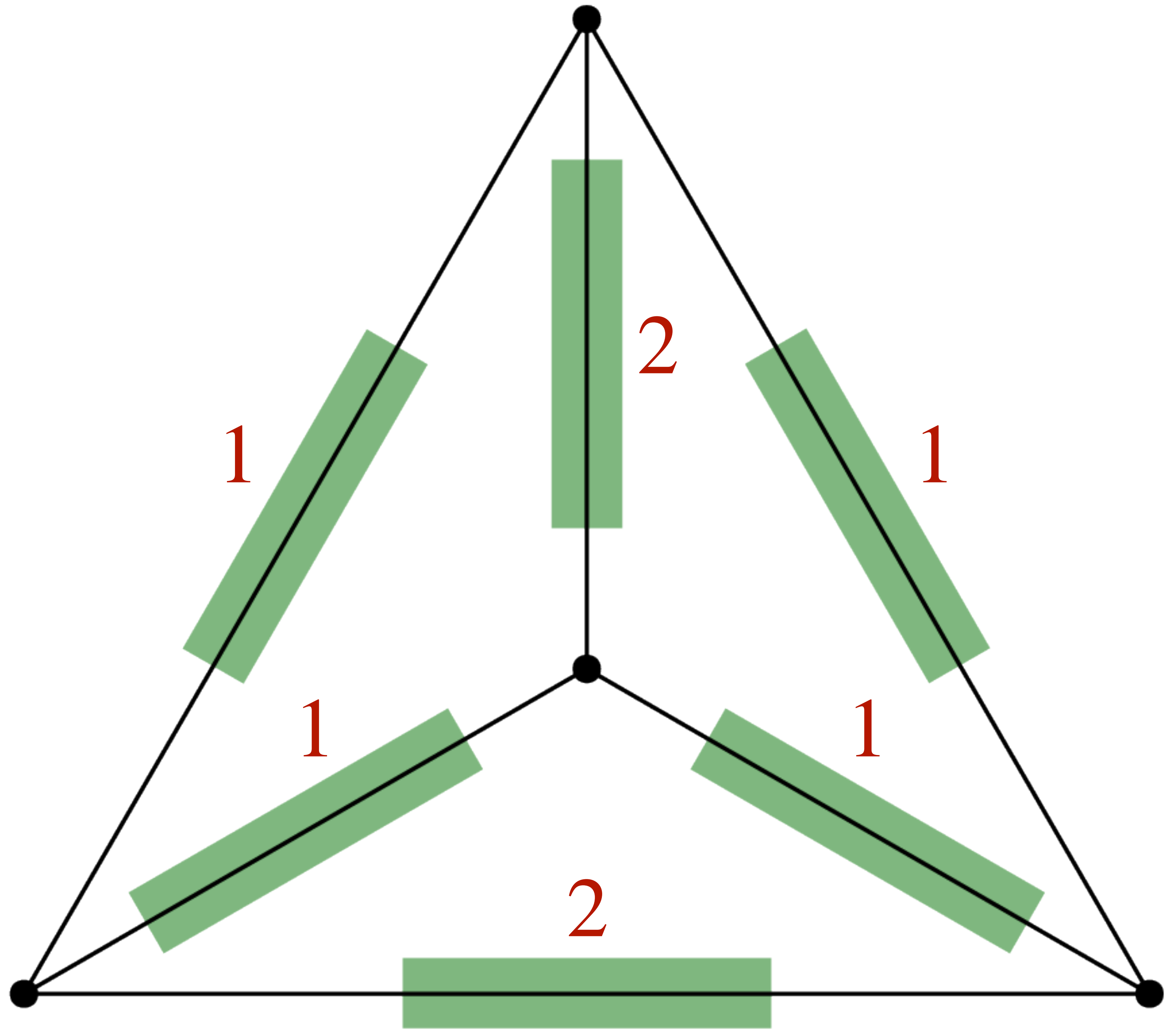


Lemma. We can construct a d-vertex tree with degrees $x_1, \ldots, x_d \geq 1$ satisfying $\sum_{i=1}^{d} x_i = 2(d-1)$ in $O(d)$ time.

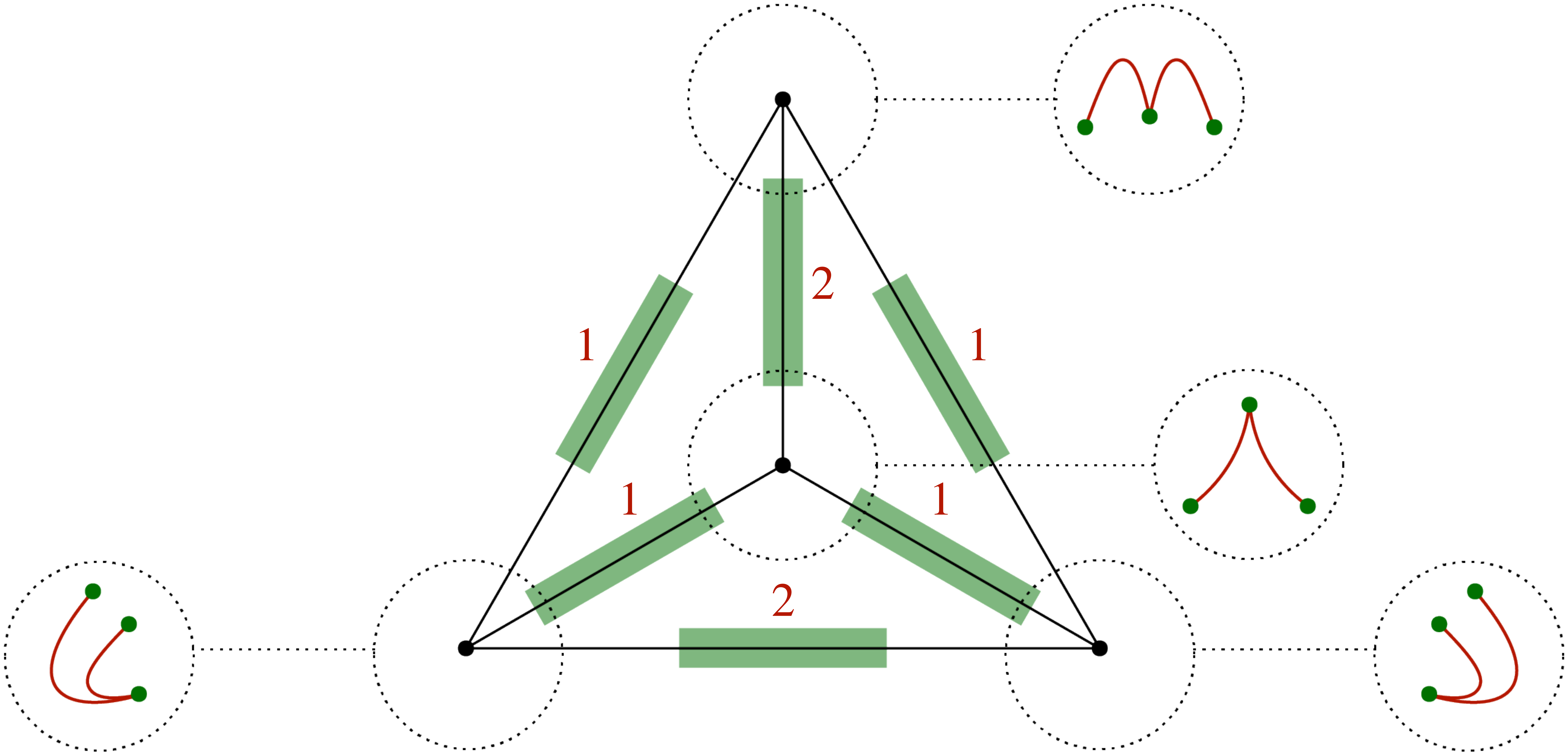
Recursion:

- Choose $x_i > 1$ and $x_j = 1$
- Connect vertices i and j
- Set $x_i = x_i - 1$, and x_j no longer exists

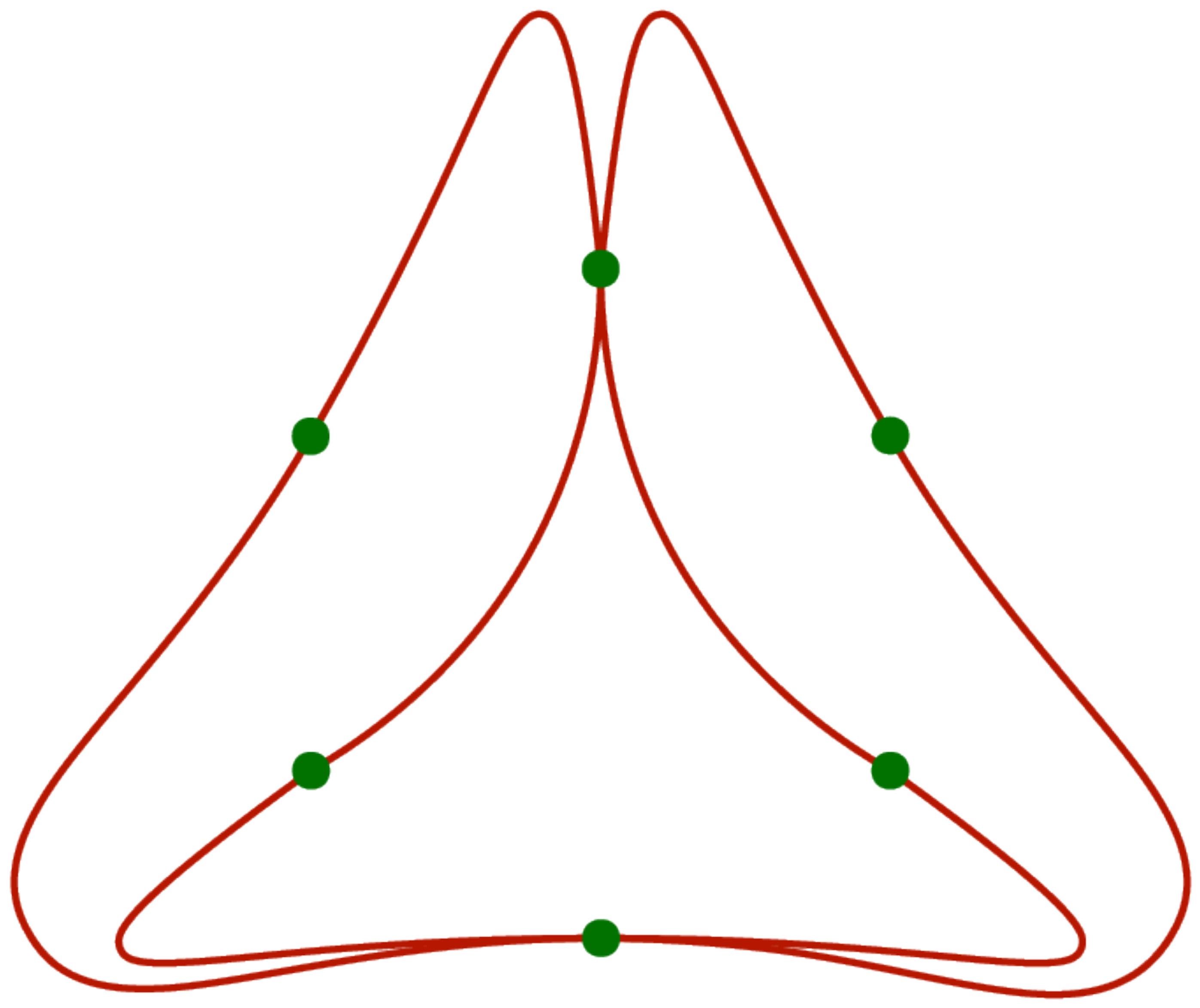
Base Case: Create a one-edge path since $d = 2$



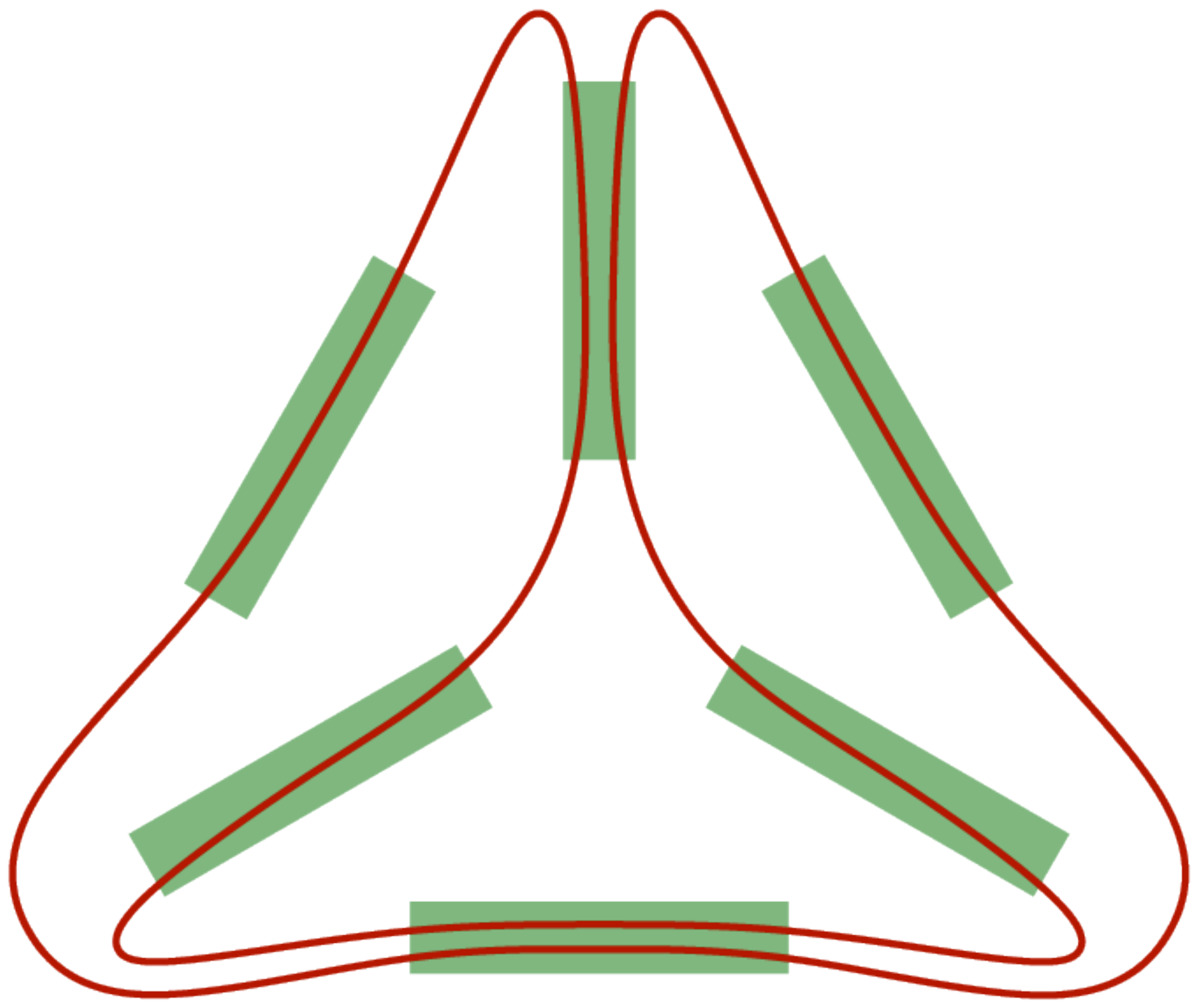
2
1
1
1
1
2


2
1
1
1
1
2

Step 2: Obtain a Threading



union of junction graphs



forbidden pattern Euler tour
[Bosboom et al. 2020]

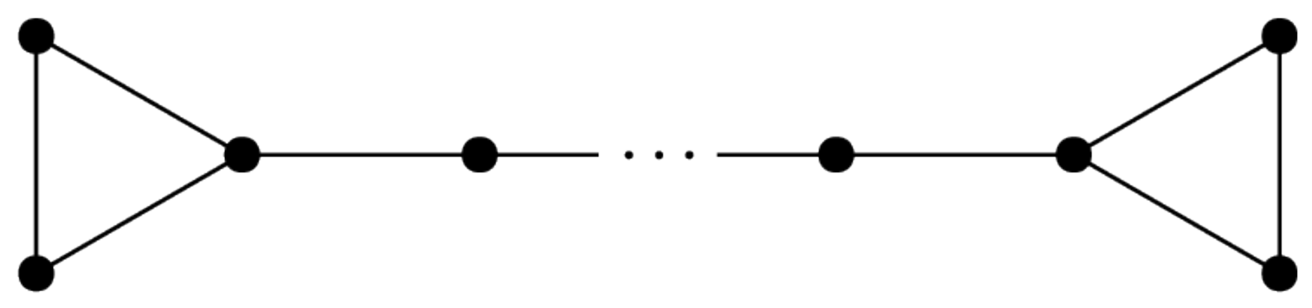
Bounds

Bounds

Recall: An optimal threading has at most length $2m$.

Bounds

Recall: An optimal threading has at most length $2m$.



Bounds

Lemma. Any threading must have length at least $2m - n$.

Bounds

Lemma. Any threading must have length at least $2m - n$.

Proof. $$\sum_{uv \in E} x_{uv} = \frac{1}{2} \sum_{v \in V} \sum_{u \in N(v)} x_{uv} \geq \frac{1}{2} \sum_{v \in V} 2(d(v) - 1) \underset{\substack{\uparrow \\ \text{handshaking}}}{=} 2m - n$$

Bounds

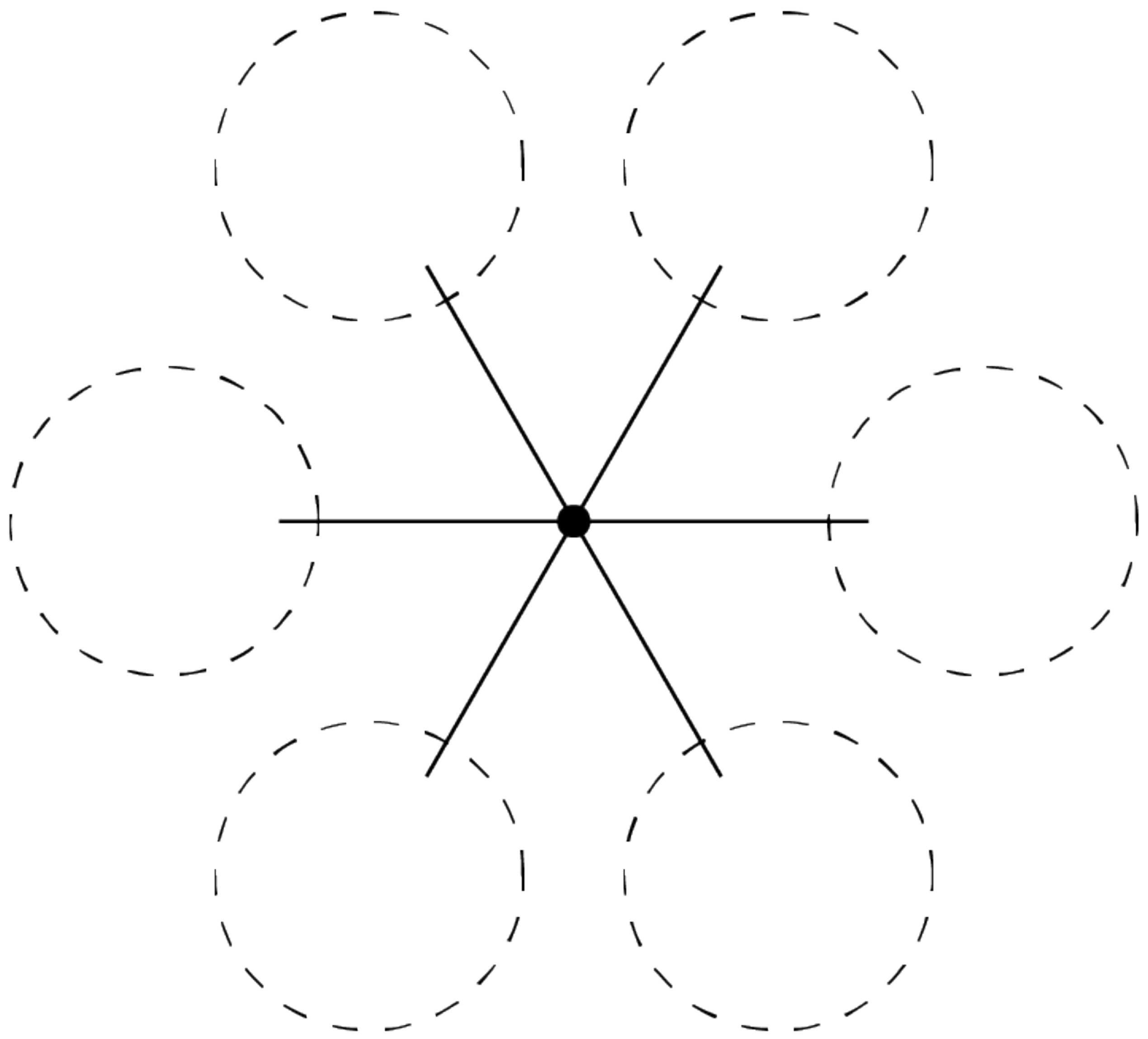
Lemma. Any threading must have length at least $2m - n$.

Definition. A **perfect threading** is a graph threading of length $2m - n$.

Bounds

Lemma. Any threading must have length at least $2m - n$.

Definition. A **perfect threading** is a graph threading of length $2m - n$.



Finding a Local Threading via Perfect Matching

Finding a Local Threading via Perfect Matching

For a perfect threading*, (C4) holds with an equality:

(C4*) $$\sum_{u \in N(v)} x_{uv} = 2(d(v) - 1) \text{ for all } v \in V$$

* Refer to paper for generalization to graphs *without* perfect threadings

Finding a Local Threading via Perfect Matching

For a perfect threading, (C4) holds with an equality:

(C4*) $$\sum_{u \in N(v)} x_{uv} = 2(d(v) - 1) \text{ for all } v \in V$$

Lemma. $\{x_{uv}\}$ is a a perfect threading if and only if it satisfies (C1) and (C4*).

Finding a Local Threading via Perfect Matching

For a perfect threading, (C4) holds with an equality:

(C4*) $\sum_{u \in N(v)} x_{uv} = 2(d(v)-1)$ for all $v \in V$

Lemma. $\{x_{uv}\}$ is a a perfect threading if and only if it satisfies (C1) and (C4*).

Proof. (C2): $\sum_{u \in N(v)} x_{uv} \pmod 2 \underset{\substack{\uparrow \\ (C^*4)}}{=} 2(d(v)-1) \pmod 2 \equiv 0$ ✓

Finding a Local Threading via Perfect Matching

For a perfect threading, (C4) holds with an equality:

(C4*) $\sum_{u \in N(v)} x_{uv} = 2(d(v) - 1)$ for all $v \in V$

Lemma. $\{x_{uv}\}$ is a a perfect threading if and only if it satisfies (C1) and (C4*).

Proof. (C2): $\sum_{u \in N(v)} x_{uv} \pmod 2 \underset{\substack{\uparrow \\ (C4^*)}}{=} 2(d(v) - 1) \pmod 2 \equiv 0$ ✓

(C3): Rewrite (C4*) as $x_{uv} + \sum_{w \in N(v) \setminus \{x_{wv}\}} x_{wv} = 2(d(v) - 1)$

Finding a Local Threading via Perfect Matching

For a perfect threading, (C4) holds with an equality:

$$(\text{C4}^*) \quad \sum_{u \in N(v)} x_{uv} = 2(d(v)-1) \text{ for all } v \in V$$

Lemma. $\{x_{uv}\}$ is a a perfect threading if and only if it satisfies (C1) and (C4*).

Proof. (C2): $\displaystyle\sum_{u \in N(v)} x_{uv} \pmod 2 \underset{(\text{C4}^*)}{\overset{}{=}} 2(d(v)-1) \pmod 2 \equiv 0$ ✓

(C3): Rewrite (C4*) as $x_{uv} + \displaystyle\sum_{w \in N(v)\setminus\{x_{wv}\}} x_{wv} = 2(d(v)-1)$

$\geq d(v)-1$ by (C1)

Finding a Local Threading via Perfect Matching

For a perfect threading, (C4) holds with an equality:

$$\text{(C4}^*\text{)} \quad \sum_{u \in N(v)} x_{uv} = 2(d(v) - 1) \text{ for all } v \in V$$

Lemma. $\{x_{uv}\}$ is a a perfect threading if and only if it satisfies (C1) and (C4*).

Proof. (C2): $\displaystyle\sum_{u \in N(v)} x_{uv} \pmod 2 \underset{\substack{\uparrow \\ (\text{C4}^*)}}{=} 2(d(v) - 1) \pmod 2 \equiv 0$ ✓

(C3): Rewrite (C4*) as $\displaystyle x_{uv} + \sum_{w \in N(v) \setminus \{x_{wv}\}} x_{wv} = 2(d(v) - 1)$ ✓

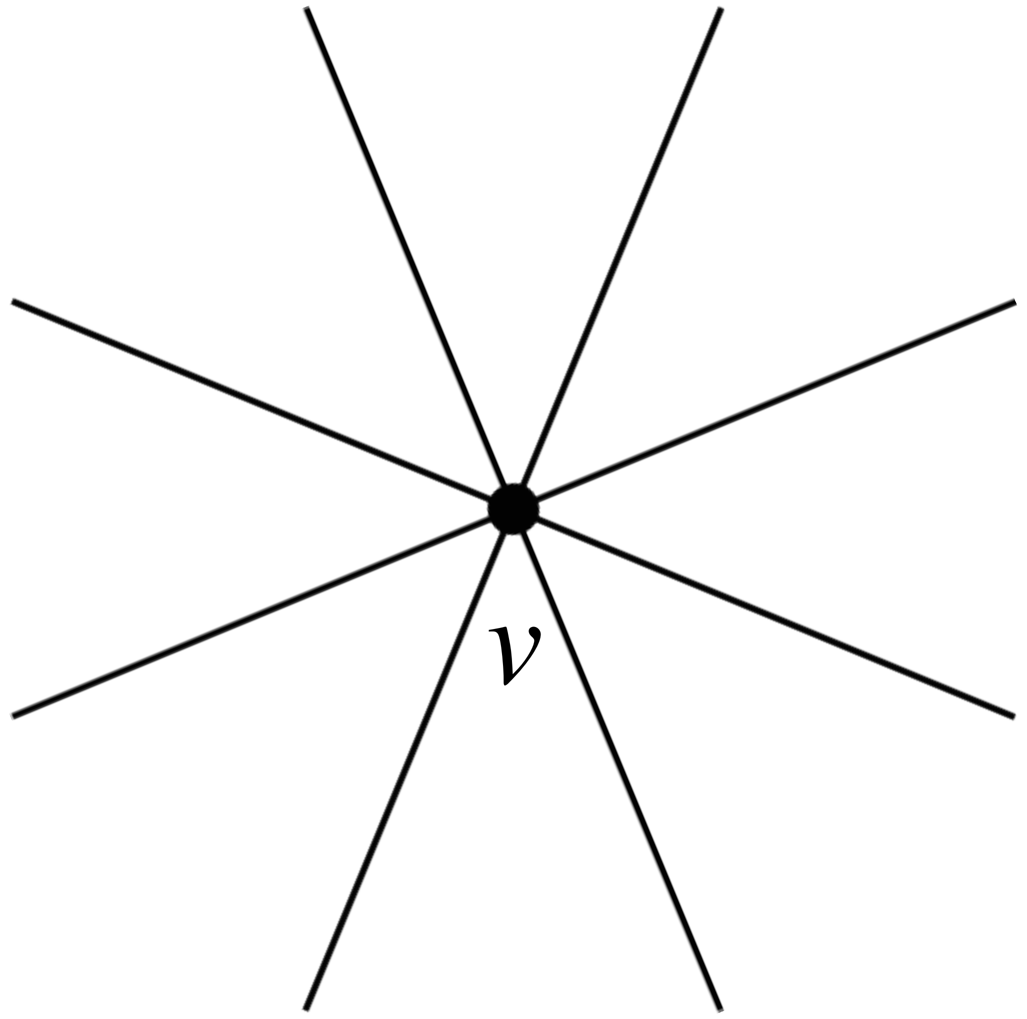
$\geq d(v) - 1$ by (C1)

Finding a Local Threading via Perfect Matching

For a perfect threading, (C4) holds an an equality:

(C4*) $$\sum_{u \in N(v)} x_{uv} = 2(d(v) - 1) \text{ for all } v \in V$$

Lemma. $\{x_{uv}\}$ is a a perfect threading if and only if it satisfies (C1) and (C4*).

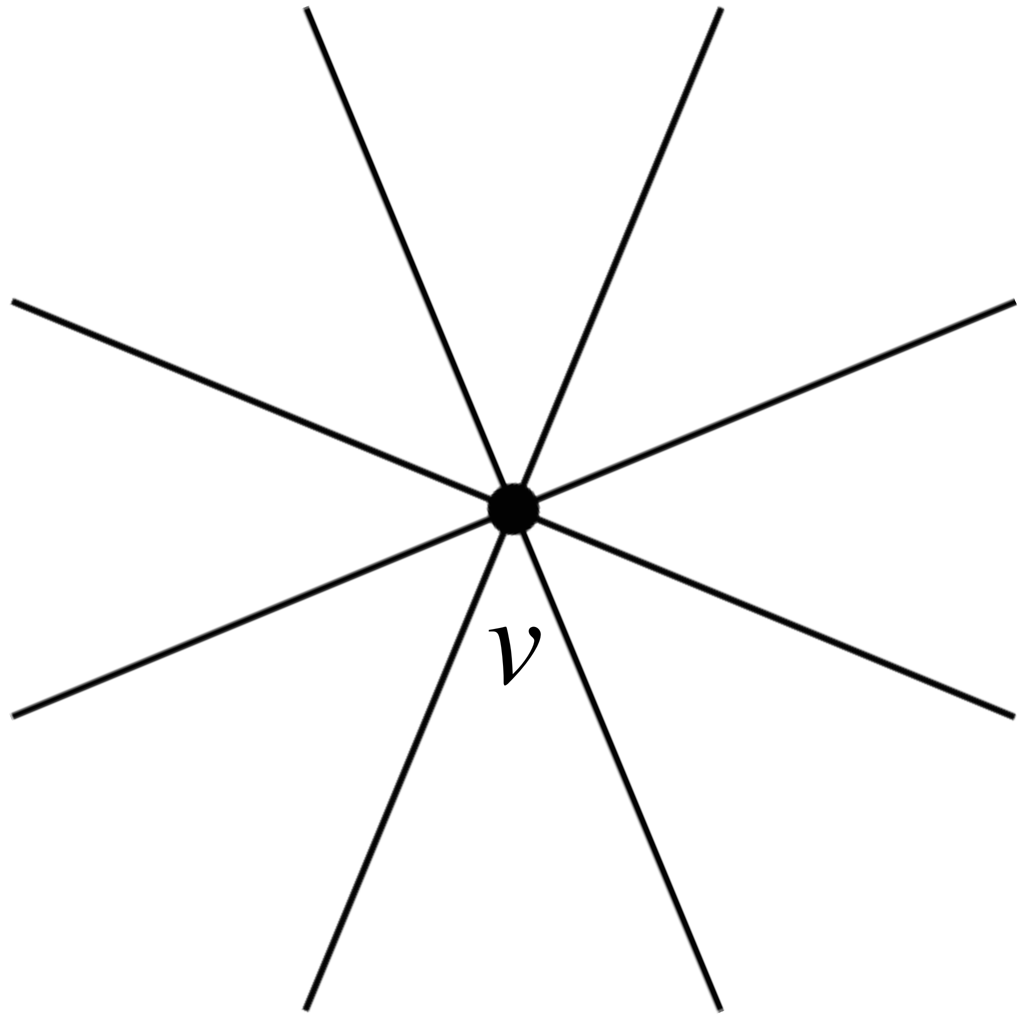


Finding a Local Threading via Perfect Matching

For a perfect threading, (C4) holds an an equality:

(C4*) $\sum_{u \in N(v)} x_{uv} = 2(d(v) - 1)$ for all $v \in V$

Lemma. $\{x_{uv}\}$ is a a perfect threading if and only if it satisfies (C1) and (C4*).



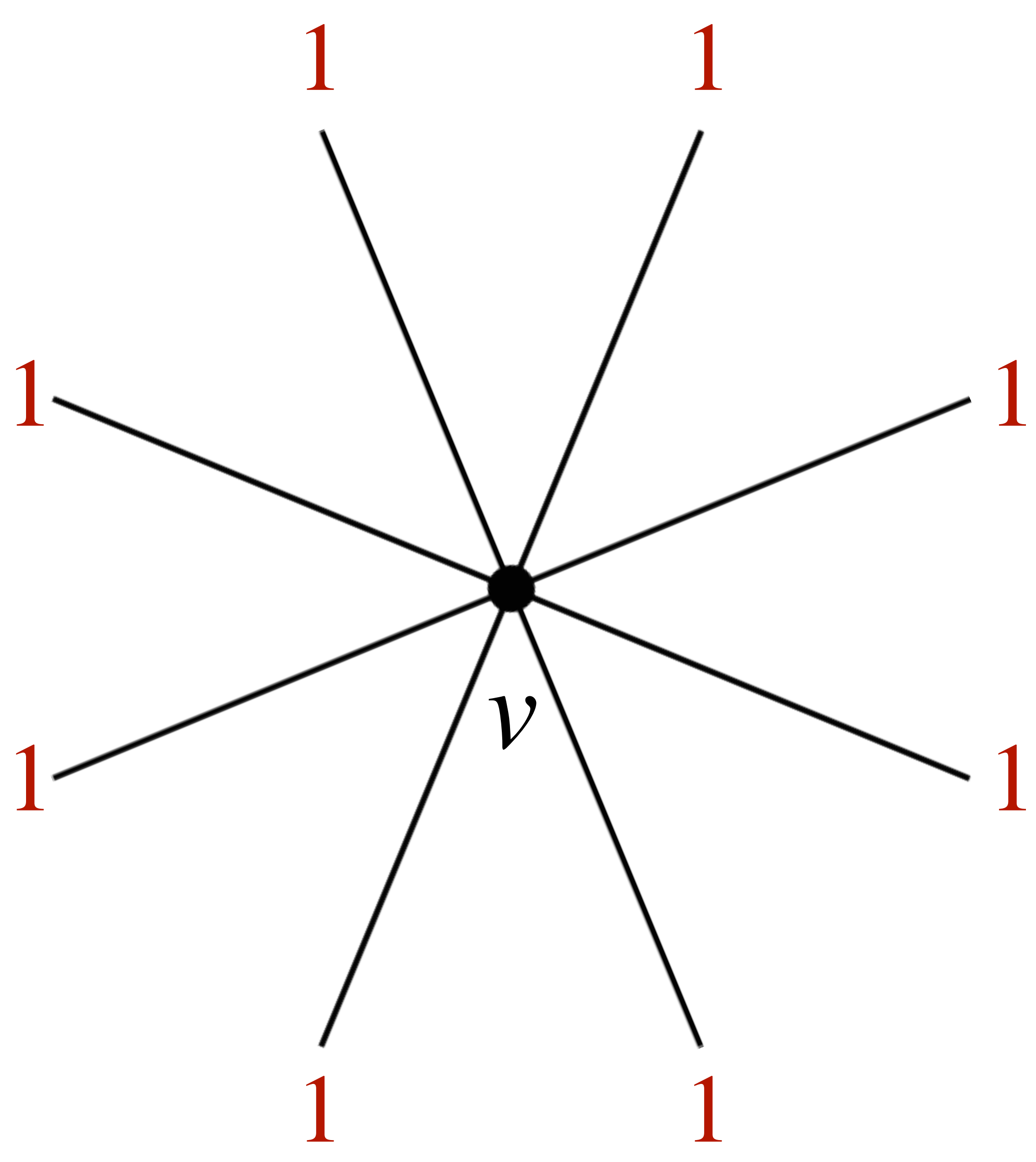
How do we distribute $2(d(v) - 1)$ units amongst the edges incident to v?

Finding a Local Threading via Perfect Matching

For a perfect threading, (C4) holds an an equality:

$$(\text{C4}^*) \quad \sum_{u \in N(v)} x_{uv} = 2(d(v) - 1) \text{ for all } v \in V$$

Lemma. $\{x_{uv}\}$ is a a perfect threading if and only if it satisfies (C1) and (C4*).



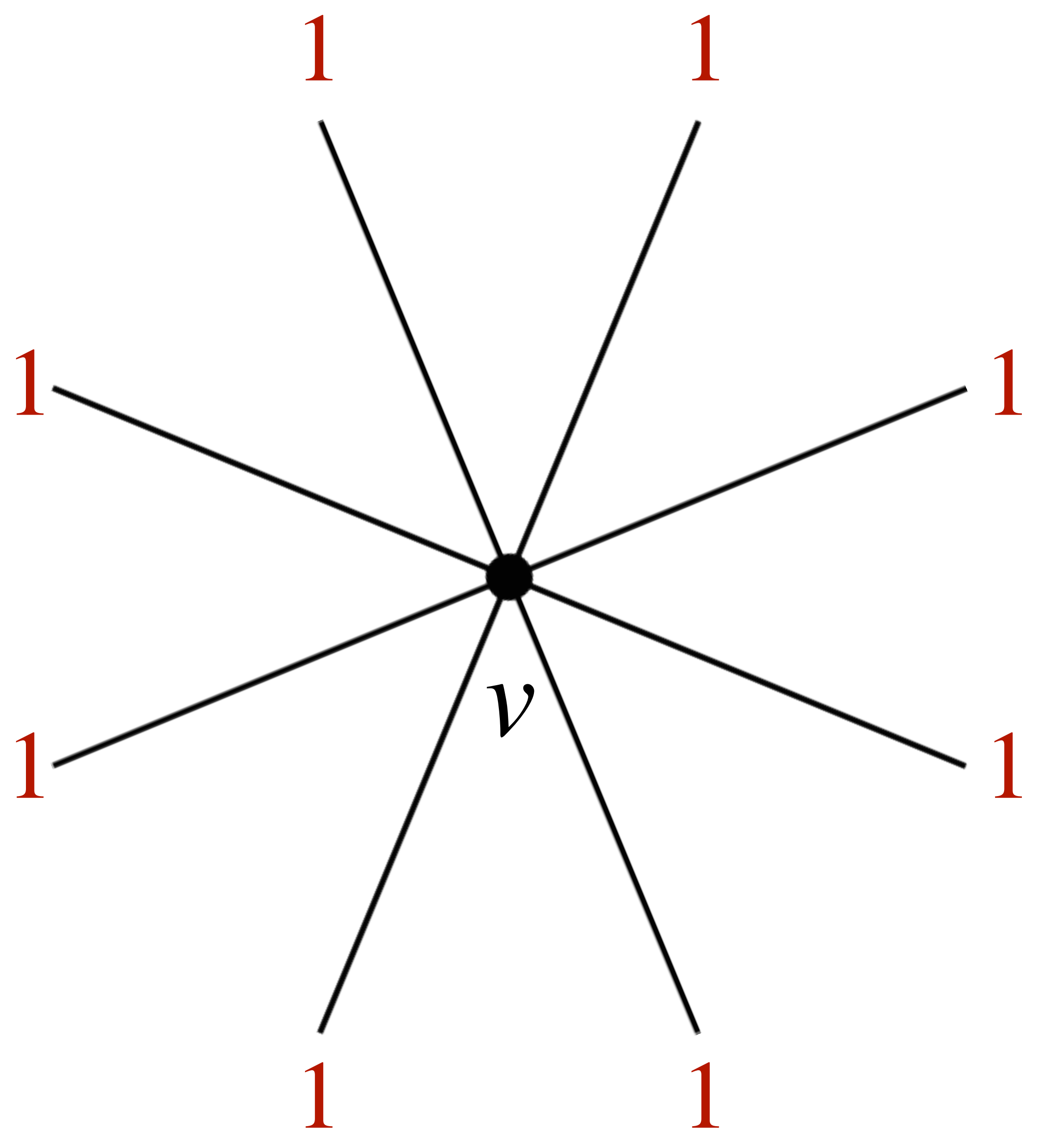
How do we distribute $2(d(v) - 1)$ units amongst the edges incident to v?

Finding a Local Threading via Perfect Matching

For a perfect threading, (C4) holds an an equality:

$$(\text{C4}^*) \quad \sum_{u \in N(v)} x_{uv} = 2(d(v) - 1) \text{ for all } v \in V$$

Lemma. $\{x_{uv}\}$ is a a perfect threading if and only if it satisfies (C1) and (C4*).

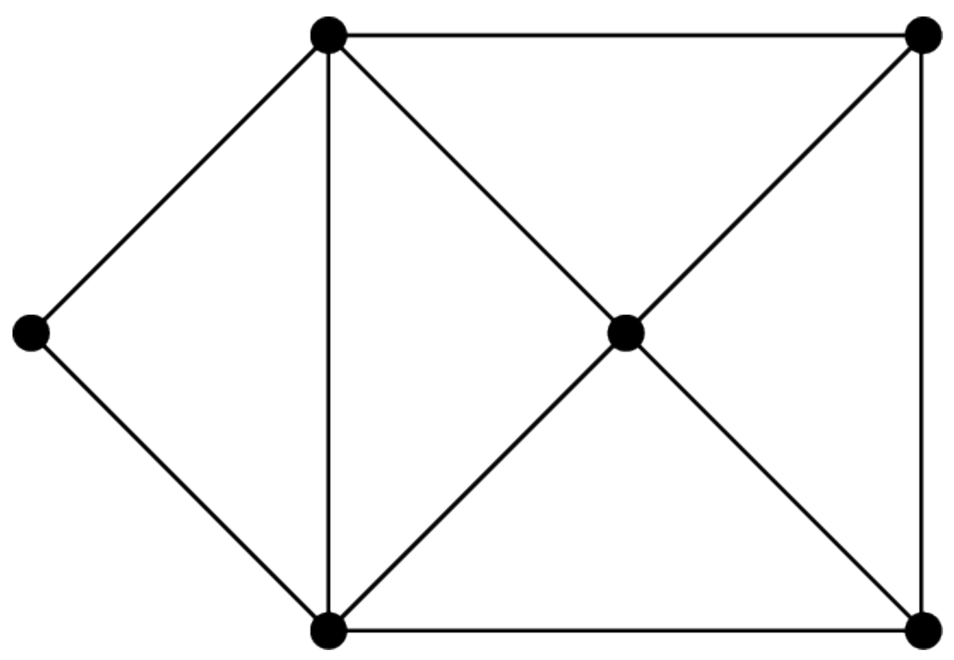


How do we distribute the *remaining* $d(v) - 2$ units amongst the edges incident to v?

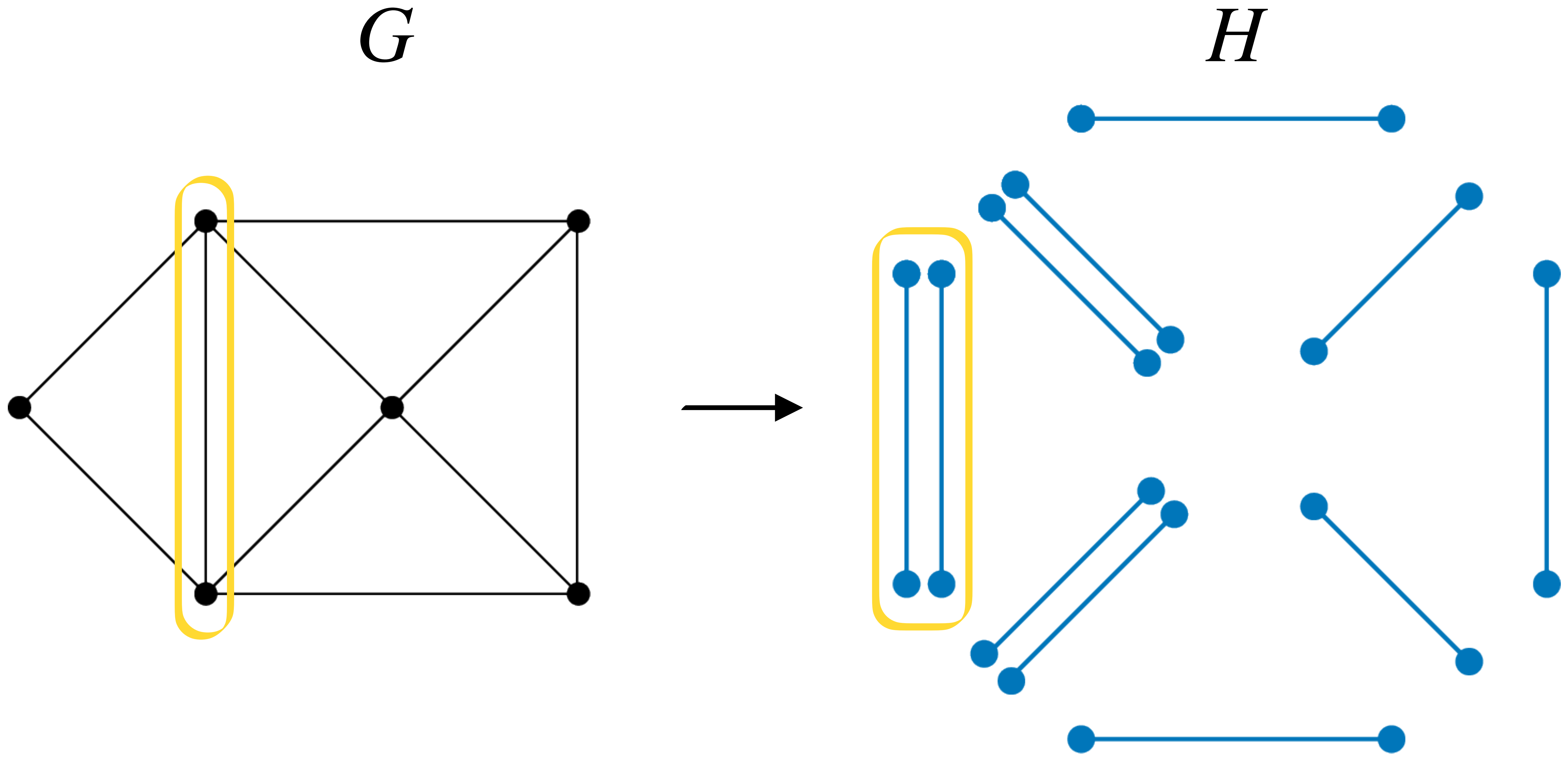
Approach: Construct a graph H that has a perfect matching if and only if, for every vertex v, we can distribute $d(v) - 2$ units among its incident edges.

Approach: Construct a graph H that has a perfect matching if and only if, for every vertex v, we can distribute $d(v) - 2$ units among its incident edges.

G

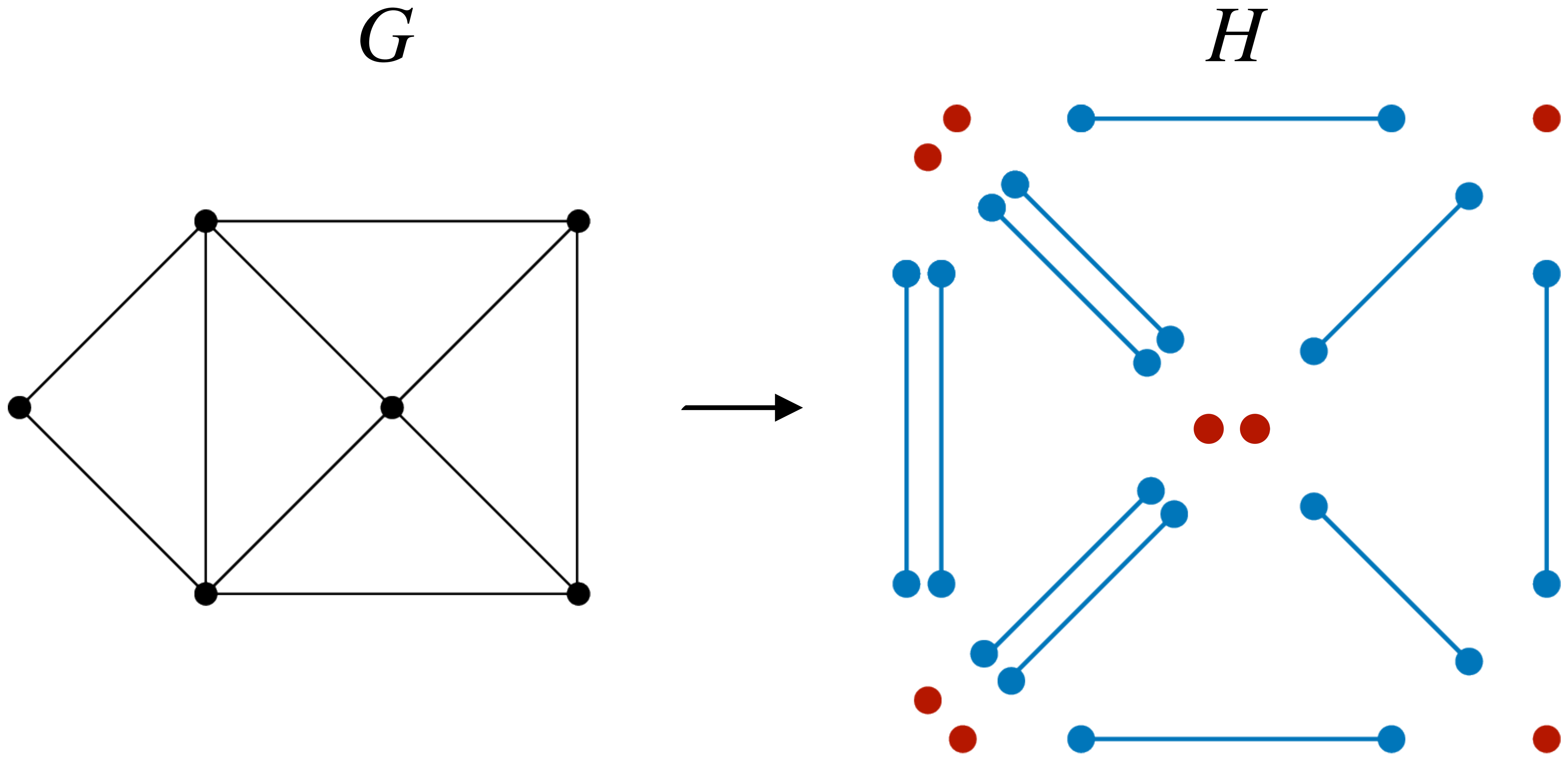


Approach: Construct a graph H that has a perfect matching if and only if, for every vertex v, we can distribute $d(v) - 2$ units among its incident edges.



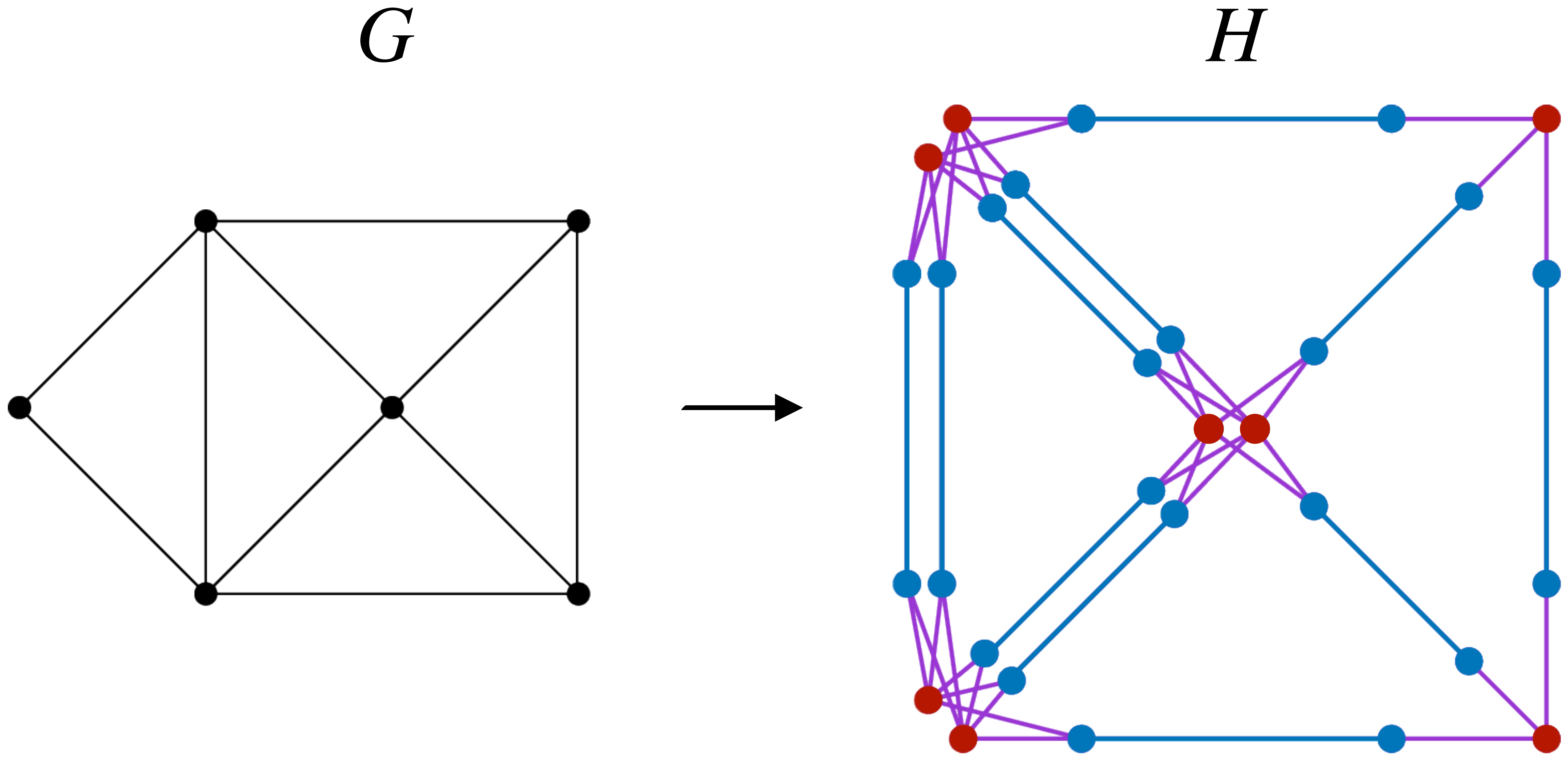
Step 1: For each $uv \in E$, create $d_{uv} := \min\{d(u), d(v)\} - 2$ disjoint edges

Approach: Construct a graph H that has a perfect matching if and only if, for every vertex v, we can distribute $d(v) - 2$ units among its incident edges.



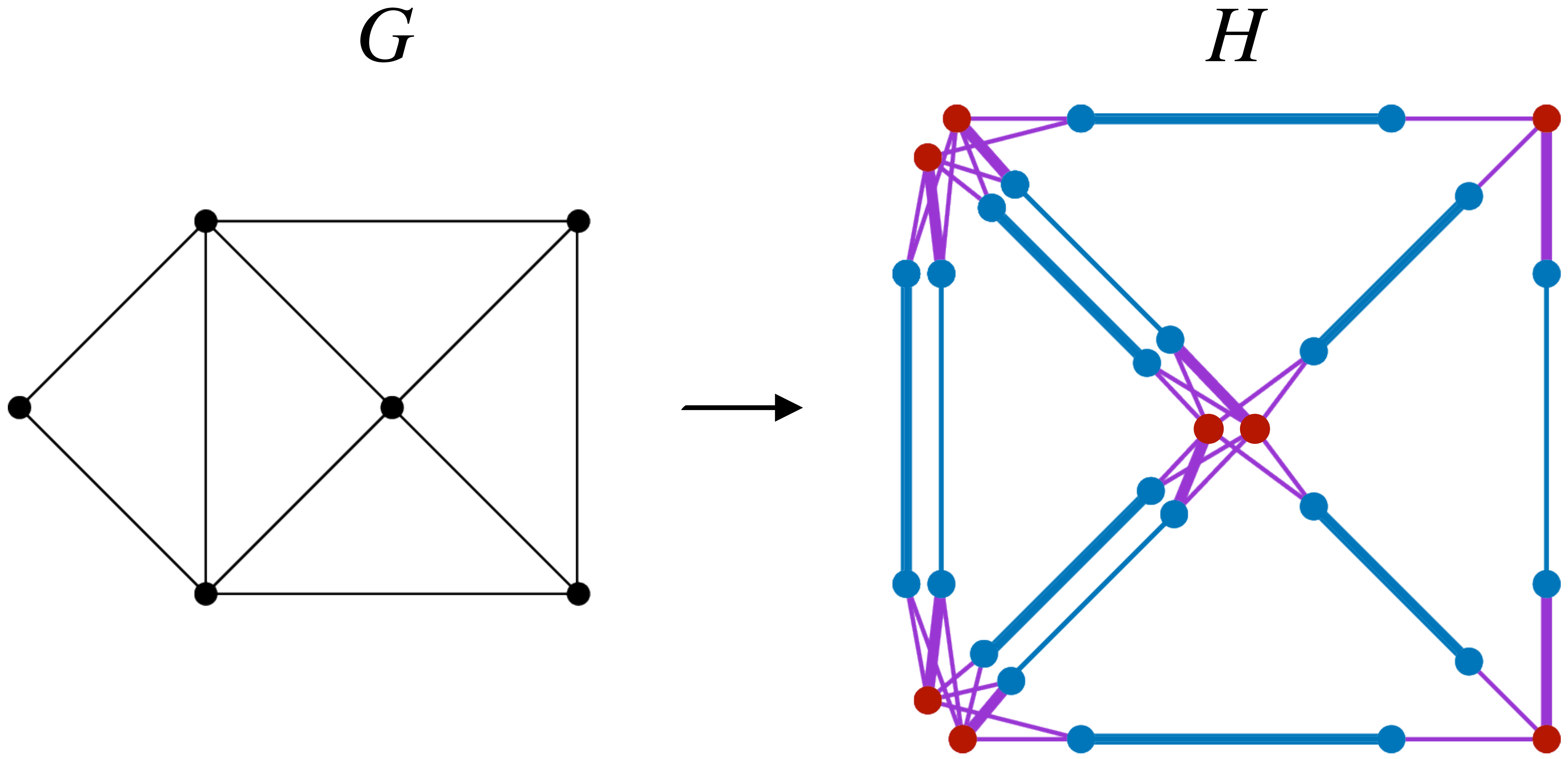
Step ~~2~~: For each vertex v, create $d(v) - 2$ vertices

Approach: Construct a graph H that has a perfect matching if and only if, for every vertex v, we can distribute $d(v) - 2$ units among its incident edges.



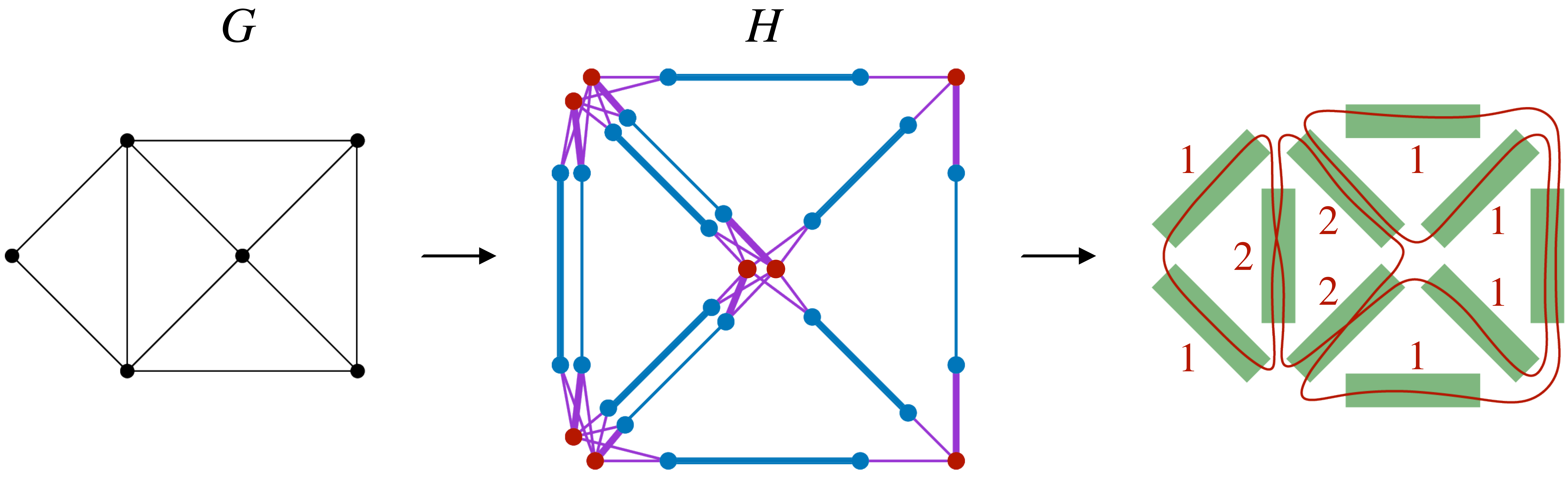
Step 3: Form bicliques at each junction

Approach: Construct a graph H that has a perfect matching if and only if, for every vertex v, we can distribute $d(v) - 2$ units among its incident edges.



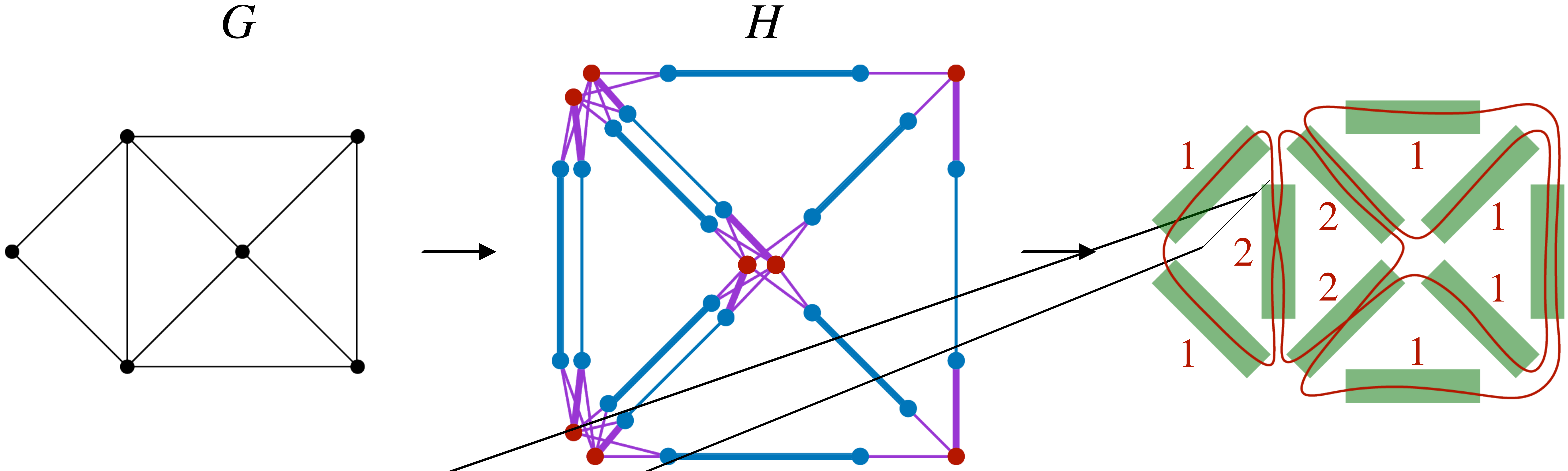
$x_{uv} := 1 \ + $ (# of blue "uv"s *not* in M), where $M \subseteq E(H)$ is a perfect matching of H

Approach: Construct a graph H that has a perfect matching if and only if, for every vertex v, we can distribute $d(v) - 2$ units among its incident edges.



$x_{uv} := 1\ + $ (# of blue "uv"s *not* in M), where $M \subseteq E(H)$ is a perfect matching of H

Approach: Construct a graph H that has a perfect matching if and only if, for every vertex v, we can distribute $d(v) - 2$ units among its incident edges.



$x_{uv} := 1 + $(# of blue "$uv$"s *not* in M), where $M \subseteq E(H)$ is a perfect matching of H

Theorem. G has a perfect threading if and only if H has a perfect matching.

Next Steps

Next Steps

- Developer tighter bounds dependent on properties of the input graph

Next Steps

- Developer tighter bounds dependent on properties of the input graph
- Devise a more efficient solution to the general problem

Next Steps

- Developer tighter bounds dependent on properties of the input graph
- Devise a more efficient solution to the general problem
- Investigate angular metric graph threading

thank you

thank you