Bayesian Probing on Graphs

Anupam Gupta — New York University

Ben Moseley — Carnegie Mellon

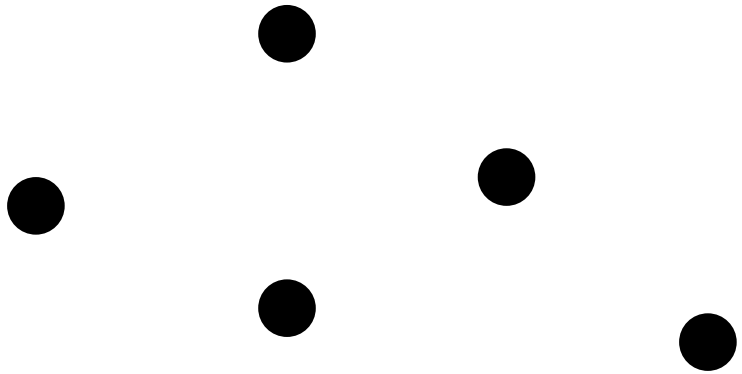
Rudy Zhou — Microsoft

Active Search

- Find objects of interest with limited budget

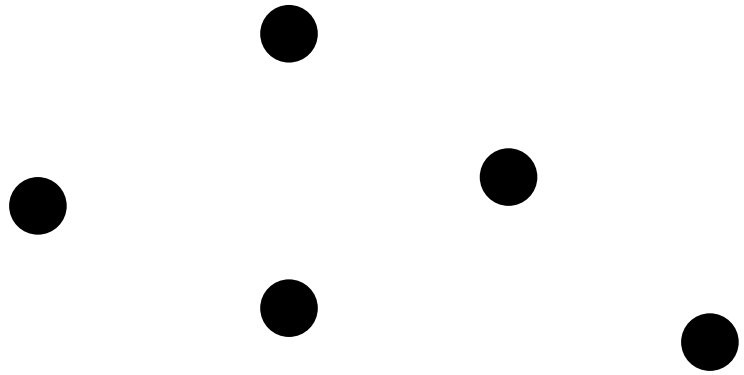
Active Search

- Find objects of interest with limited budget
- **Input:**
  - $U =$ universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets



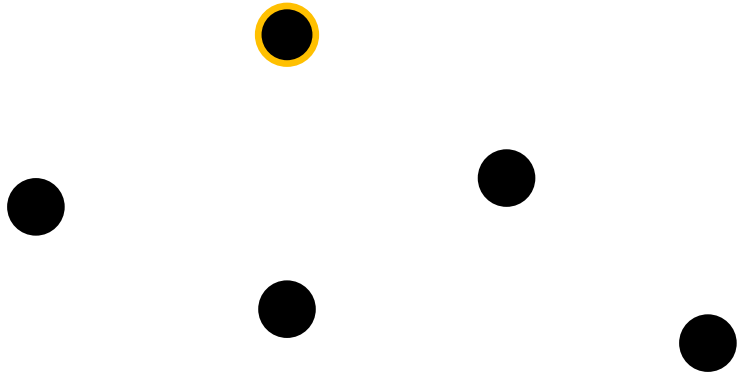
Active Search

- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$



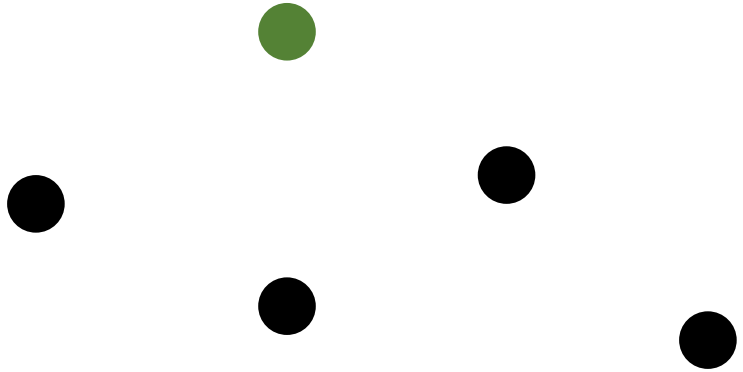
Active Search

- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes (‘ $x \in T$ or not’)
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$



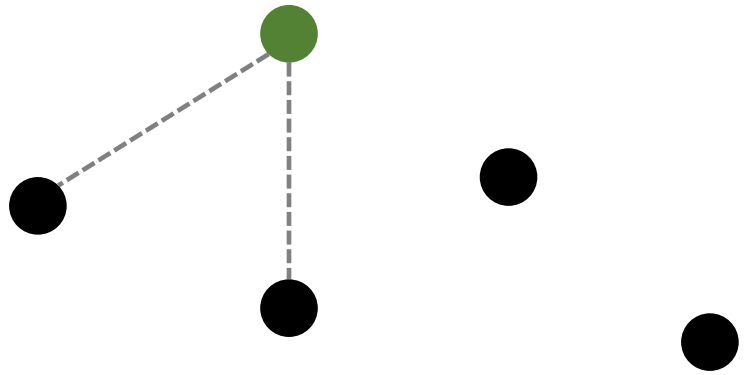
Active Search

- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes (‘ $x \in T$ or not’)
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$



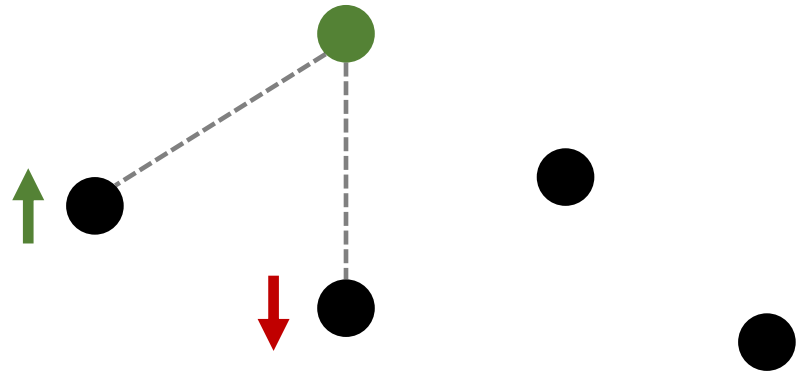
Active Search

- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes (‘ $x \in T$ or not’)
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$



Active Search

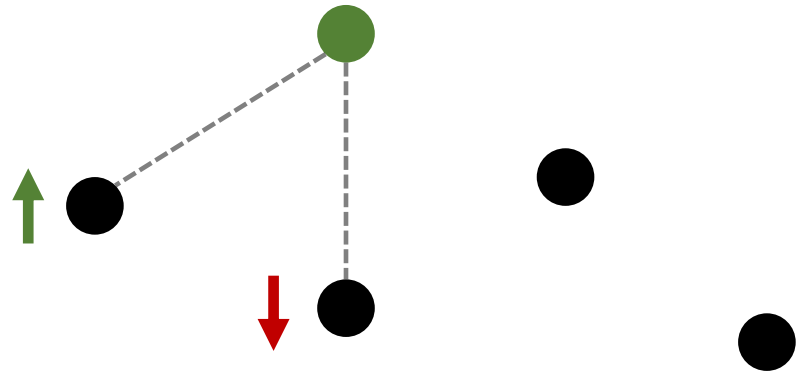
- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$



Probing elements causes Bayesian update on future tests

Active Search

- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$

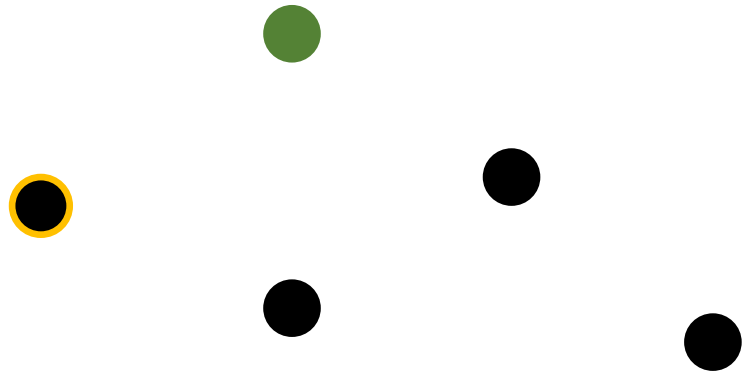


- **Goal:** Adaptively probe at most k elements, $P \subset U$ to maximize the expected number of targets found: $\mathbb{E}[\sum_{x \in P} 1(x \in T)]$

Probing elements causes Bayesian update on future tests

Active Search

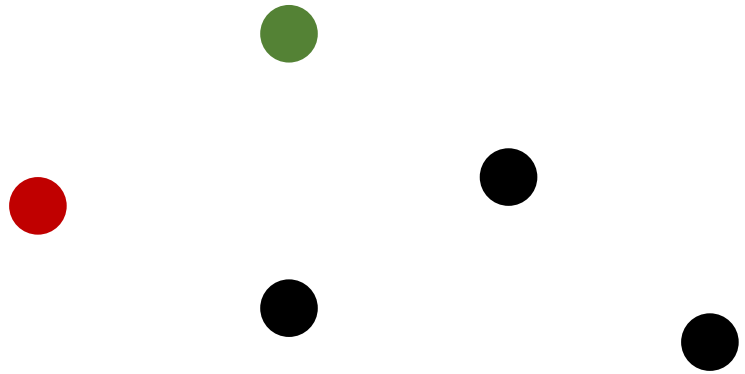
- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$
- **Goal:** Adaptively probe at most k elements, $P \subset U$ to maximize the expected number of targets found: $\mathbb{E}[\sum_{x \in P} 1(x \in T)]$



Probing elements causes Bayesian update on future tests

Active Search

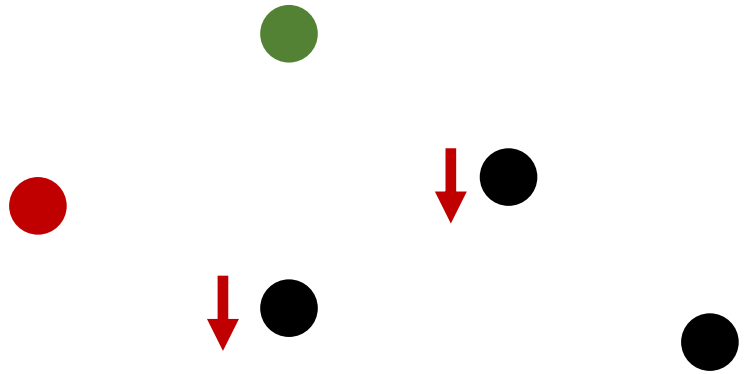
- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$
- **Goal:** Adaptively probe at most k elements, $P \subset U$ to maximize the expected number of targets found: $\mathbb{E}[\sum_{x \in P} 1(x \in T)]$



Probing elements causes Bayesian update on future tests

Active Search

- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$

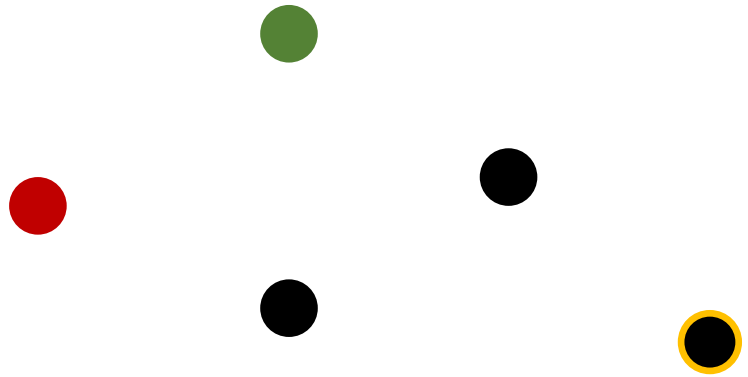


- **Goal:** Adaptively probe at most k elements, $P \subset U$ to maximize the expected number of targets found: $\mathbb{E}[\sum_{x \in P} 1(x \in T)]$

Probing elements causes Bayesian update on future tests

Active Search

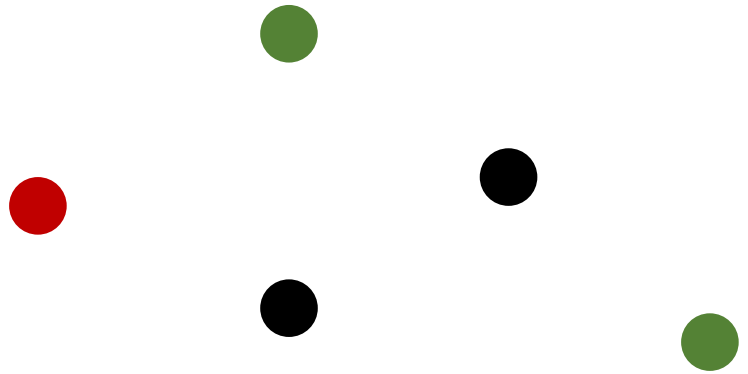
- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$
- **Goal:** Adaptively probe at most k elements, $P \subset U$ to maximize the expected number of targets found: $\mathbb{E}[\sum_{x \in P} 1(x \in T)]$



Probing elements causes Bayesian update on future tests

Active Search

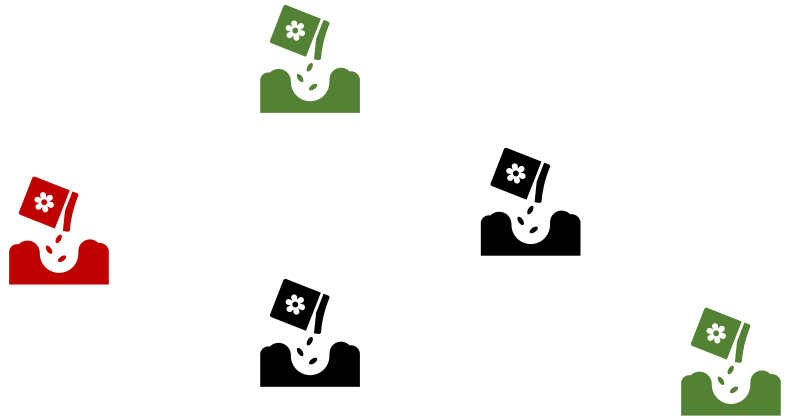
- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$
- **Goal:** Adaptively probe at most k elements, $P \subset U$ to maximize the expected number of targets found: $\mathbb{E}[\sum_{x \in P} 1(x \in T)]$



Probing elements causes Bayesian update on future tests

Active Search

- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$

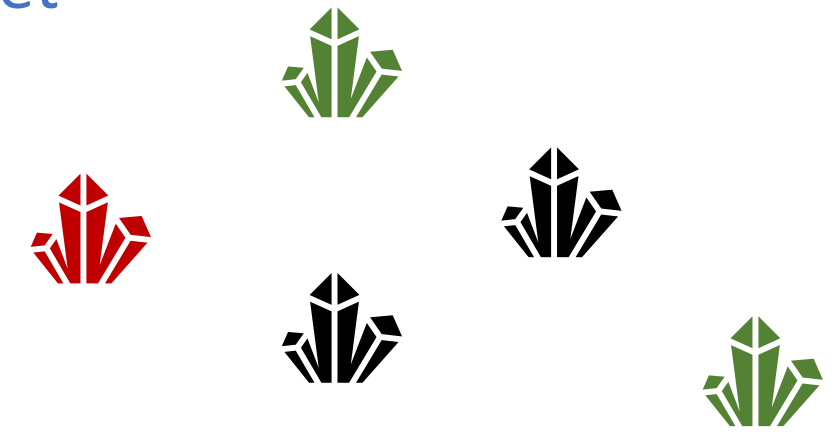


- **Goal:** Adaptively probe at most k elements, $P \subset U$ to maximize the expected number of targets found: $\mathbb{E}[\sum_{x \in P} 1(x \in T)]$

Probing elements causes Bayesian update on future tests

Active Search

- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$



- **Goal:** Adaptively probe at most k elements, $P \subset U$ to maximize the expected number of targets found: $\mathbb{E}[\sum_{x \in P} 1(x \in T)]$

Probing elements causes Bayesian update on future tests

Active Search

- Find objects of interest with limited budget
- **Input:**
  - U = universe of elements
    - unknown subset $T \subset U$ of targets
    - Known probability distribution over targets
  - k = budget of probes ('$x \in T$ or not')
    - Probing $x \in U$ reveals whether $x \in T$ or $x \notin T$

- **Goal:** Adaptively probe at most k elements, $P \subset U$ to maximize the expected number of targets found: $\mathbb{E}[\sum_{x \in P} 1(x \in T)]$

Probing elements causes Bayesian update on future tests

What's known

- Can compute optimal adaptive policy in exponential time

  Roman Garnett, Yamuna Krishnamurthy, Xuehan Xiong, Jeff G. Schneider, Richard P. Mann
  Bayesian Optimal Active Search and Surveying. ICML 2012

- Hard to approximate to any constant factor

  Shali Jiang, Gustavo Malkomes, Geoff Converse, Alyssa Shofner, Benjamin Moseley, Roman Garnett:
  Efficient Nonmyopic Active Search. ICML 2017

- Many heuristics (limited lookahead, proxy objectives, etc.)

What's known

- Can compute optimal adaptive policy in exponential time

  Roman Garnett, Yamuna Krishnamurthy, Xuehan Xiong, Jeff G. Schneider, Richard P. Mann
  Bayesian Optimal Active Search and Surveying. ICML 2012

- Hard to approximate to any constant factor

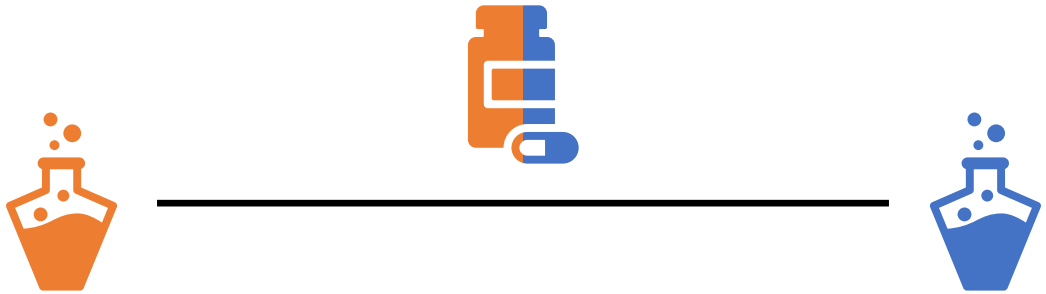
  Shali Jiang, Gustavo Malkomes, Geoff Converse, Alyssa Shofner, Benjamin Moseley, Roman Garnett:
  Efficient Nonmyopic Active Search. ICML 2017

- Many heuristics (limited lookahead, proxy objectives, etc.)

Question: Can we consider a concrete correlation model and design efficiently-computable, provably near-optimal policies?

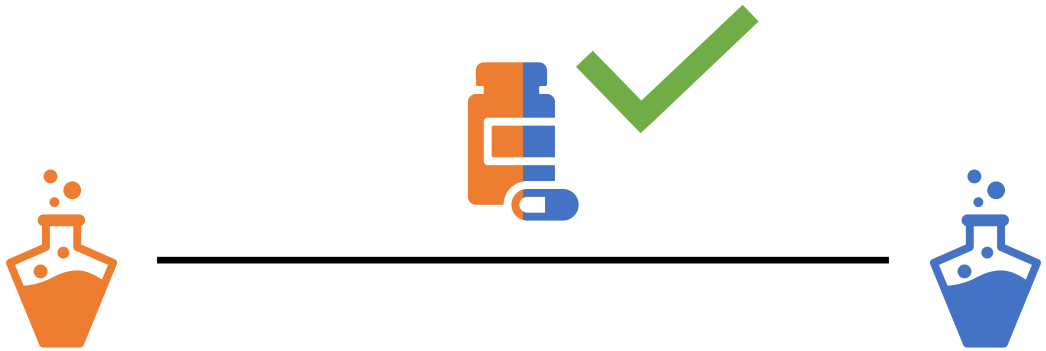
Bayesian Probing on Graphs - Motivation

- Elements consist of ingredients; elements are correlated when they share ingredients



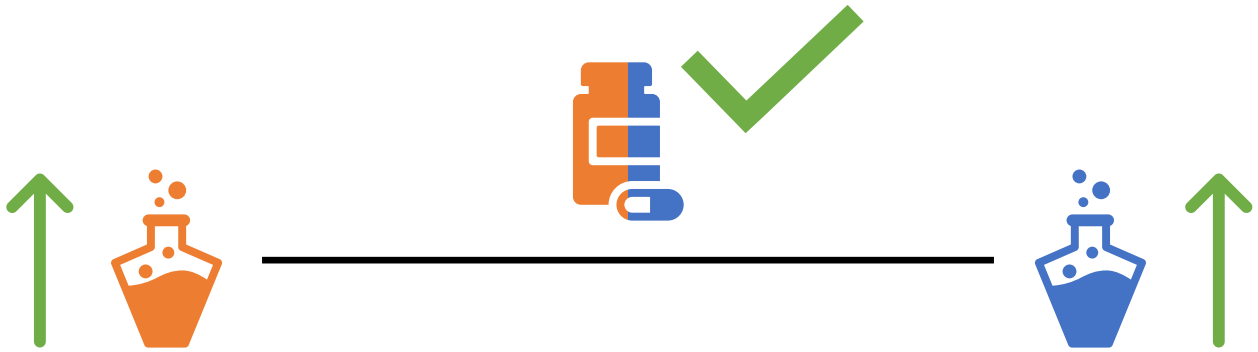
Bayesian Probing on Graphs - Motivation

- Elements consist of ingredients; elements are correlated when they share ingredients



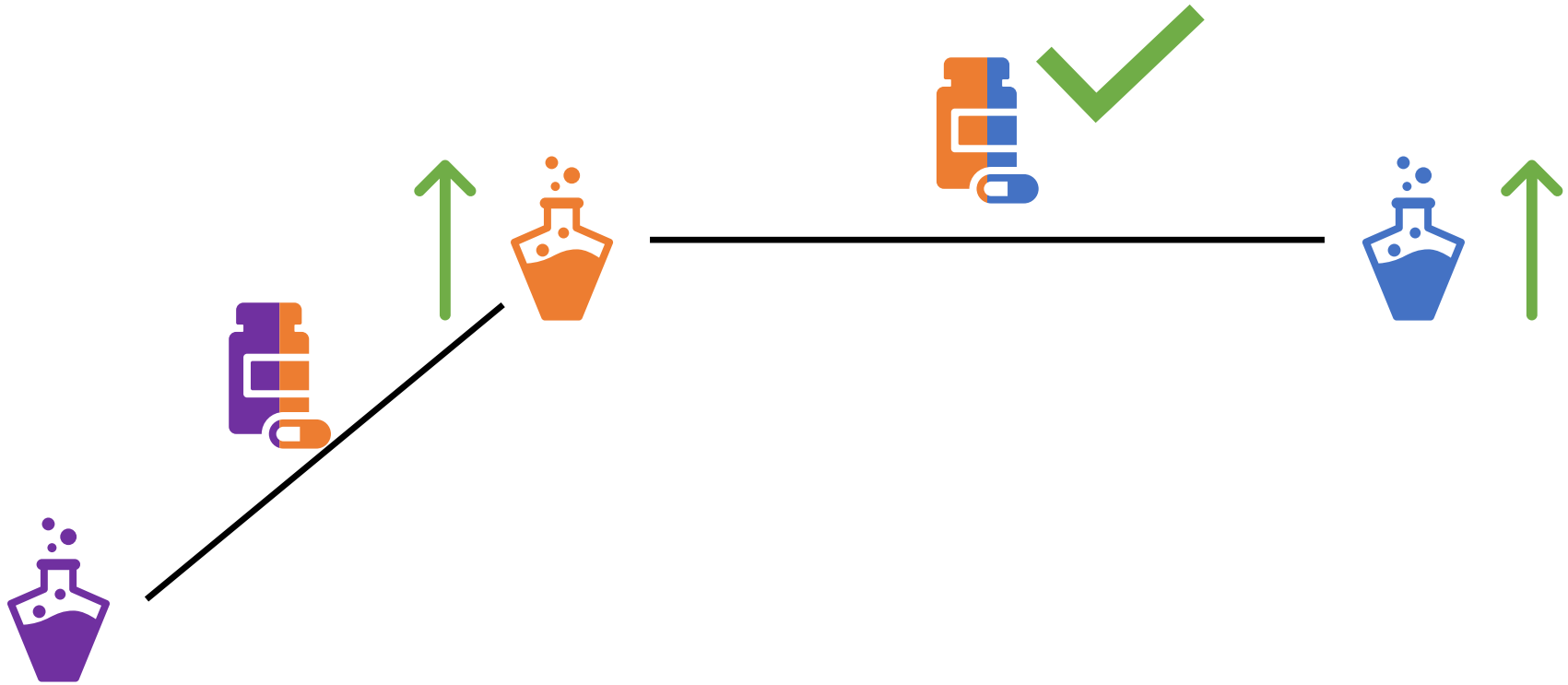
Bayesian Probing on Graphs - Motivation

- Elements consist of ingredients; elements are correlated when they share ingredients



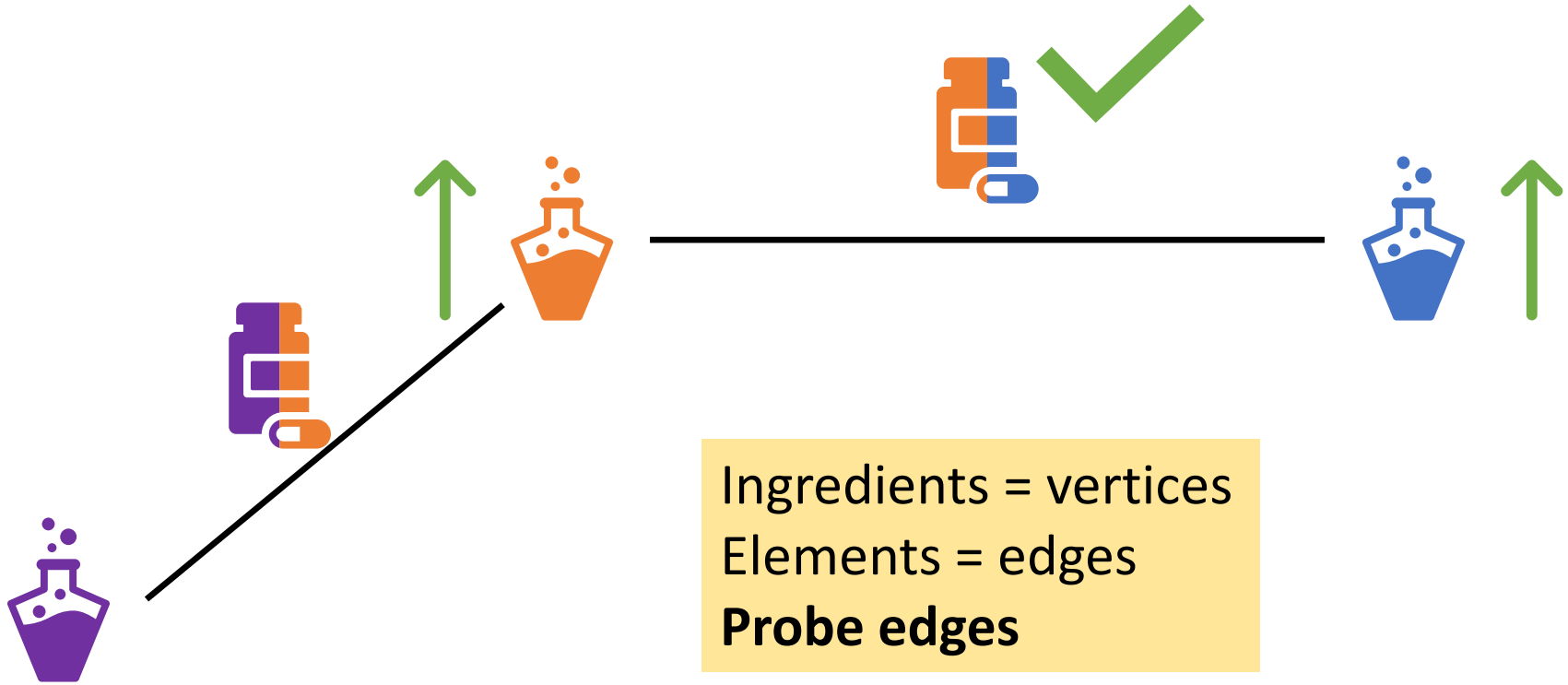
Bayesian Probing on Graphs - Motivation

- Elements consist of ingredients; elements are correlated when they share ingredients



Bayesian Probing on Graphs - Motivation

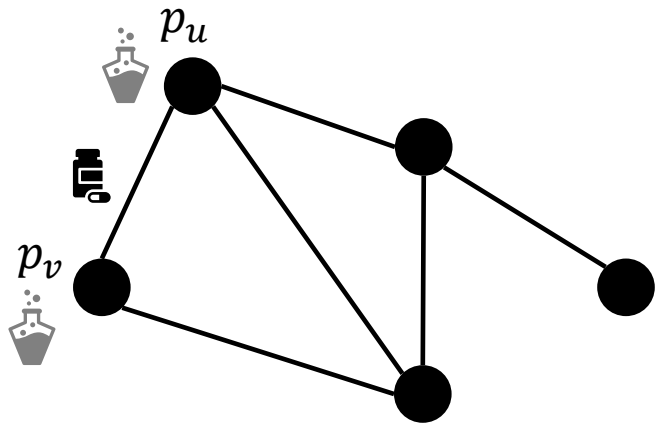
- Elements consist of ingredients; elements are correlated when they share ingredients



Ingredients = vertices
Elements = edges
Probe edges

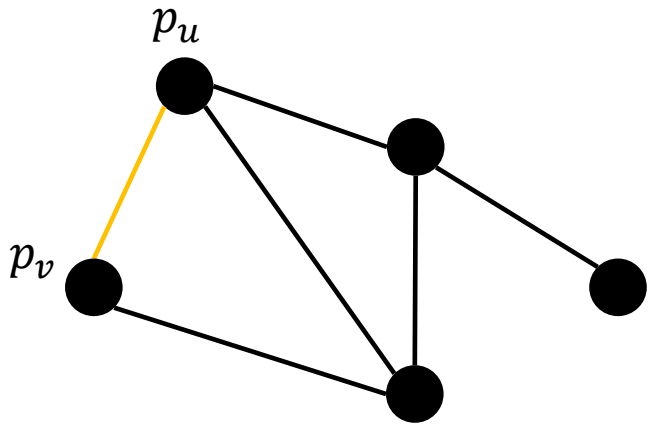
Bayesian Probing on Graphs

- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active



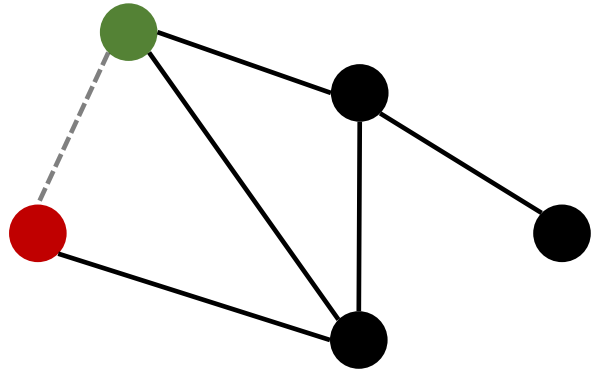
Bayesian Probing on Graphs

- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active
  - k = budget of **edge probes**
    - Probing $uv \in E$ reveals both endpoints



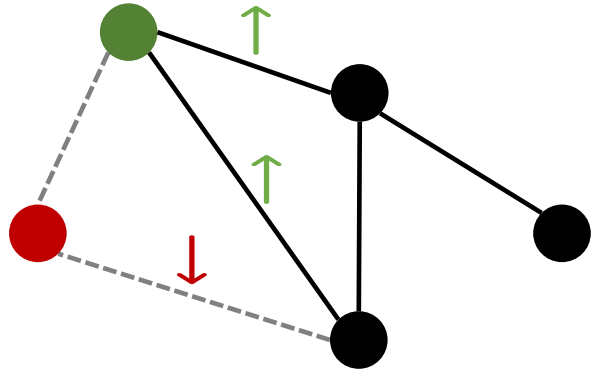
Bayesian Probing on Graphs

- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active
  - k = budget of **edge probes**
    - Probing $uv \in E$ reveals both endpoints



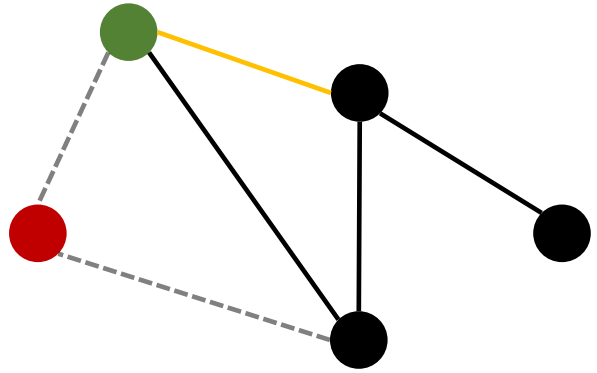
Bayesian Probing on Graphs

- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active
  - k = budget of **edge probes**
    - Probing $uv \in E$ reveals both endpoints



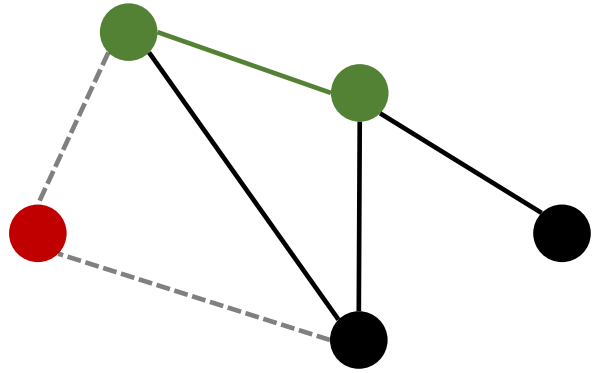
Bayesian Probing on Graphs

- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active
  - k = budget of **edge probes**
    - Probing $uv \in E$ reveals both endpoints



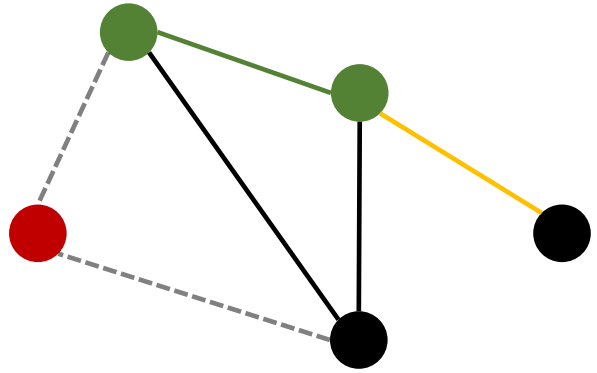
Bayesian Probing on Graphs

- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active
  - k = budget of **edge probes**
    - Probing $uv \in E$ reveals both endpoints



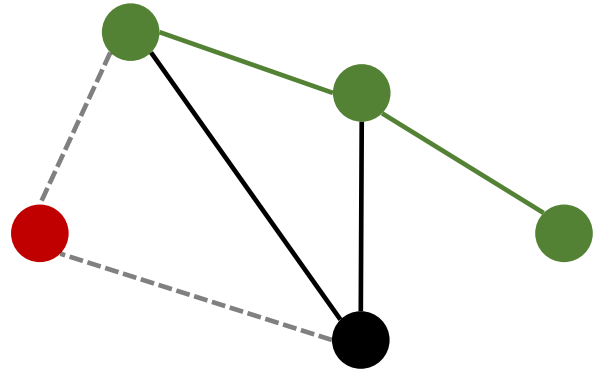
Bayesian Probing on Graphs

- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active
  - k = budget of **edge probes**
    - Probing $uv \in E$ reveals both endpoints



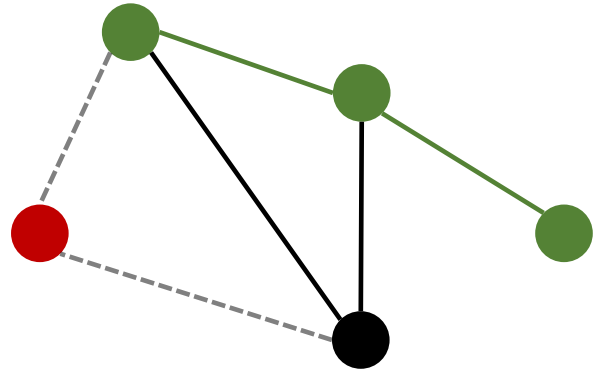
Bayesian Probing on Graphs

- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active
  - k = budget of **edge probes**
    - Probing $uv \in E$ reveals both endpoints



Bayesian Probing on Graphs

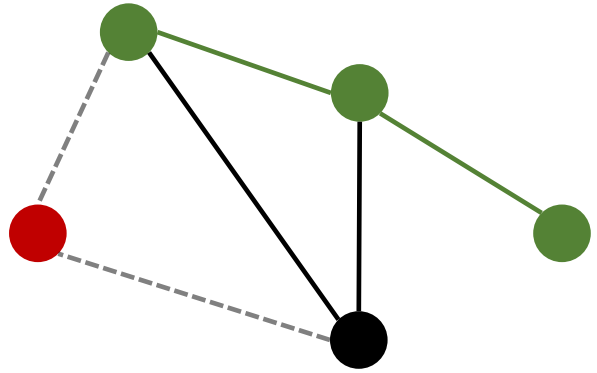
- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active
  - k = budget of **edge probes**
    - Probing $uv \in E$ reveals both endpoints



- **Goal:** Adaptively probe at most k **edges**, $P \subset E$ to maximize the expected number of active edges found: $\mathbb{E}[\sum_{e \in P} 1(e \text{ active})]$

Bayesian Probing on Graphs

- **Input:**
  - Graph $G = (V, E)$
    - Each vertex $v \in V$ has known probability p_v of being active
    - Edge $uv \in E$ is active if both of its endpoints are active
  - k = budget of **edge probes**
    - Probing $uv \in E$ reveals both endpoints



- **Goal:** Adaptively probe at most k **edges**, $P \subset E$ to maximize the expected number of active edges found: $\mathbb{E}[\sum_{e \in P} 1(e \text{ active})]$

Knapsack Generalization: Edges can have non-uniform probe rewards and costs

Connections

- Stochastic probing problems

  Anupam Gupta, Viswanath Nagarajan:
  A Stochastic Probing Problem with Applications. IPCO 2013

  Anupam Gupta, Viswanath Nagarajan, Sahil Singla:
  Adaptivity Gaps for Stochastic Probing: Submodular and XOS Functions. SODA 2017

- Stochastic knapsack with correlations across items

  Brian C. Dean, Michel X. Goemans, Jan Vondrák:
  Approximating the Stochastic Knapsack Problem: The Benefit of Adaptivity. Math. Oper. Res. 2008

  Anupam Gupta, Ravishankar Krishnaswamy, Marco Molinaro, R. Ravi:
  Approximation Algorithms for Correlated Knapsacks and Non-martingale Bandits. FOCS 2011

- Site percolation

  Broadbent SR, Hammersley JM:
  Percolation processes: I. Crystals and mazes. Mathematical Proceedings of the Cambridge Philosophical Society 1957

  Michael Krivelevich:
  The Phase Transition in Site Percolation on Pseudo-Random Graphs. Electron. J. Comb. 2016

Our Results

Question: Can we consider a concrete correlation model and design efficiently-computable, provably near-optimal policies?

Our Results

Question: Can we consider a concrete correlation model and design efficiently-computable, provably near-optimal policies?

- **Non-adaptive algorithms:** Fix up-front all edge probes
  - $O(max\ degree)$-approximation
  - Adaptivity gap is $\Omega(max\ degree)$

Our Results

Question: Can we consider a concrete correlation model and design efficiently-computable, provably near-optimal policies?

- **Non-adaptive algorithms:** Fix up-front all edge probes
  - $O(max\ degree)$-approximation
  - Adaptivity gap is $\Omega(max\ degree)$
- **Adaptive algorithms:**
  - $O(density)$-approximation

Our Results

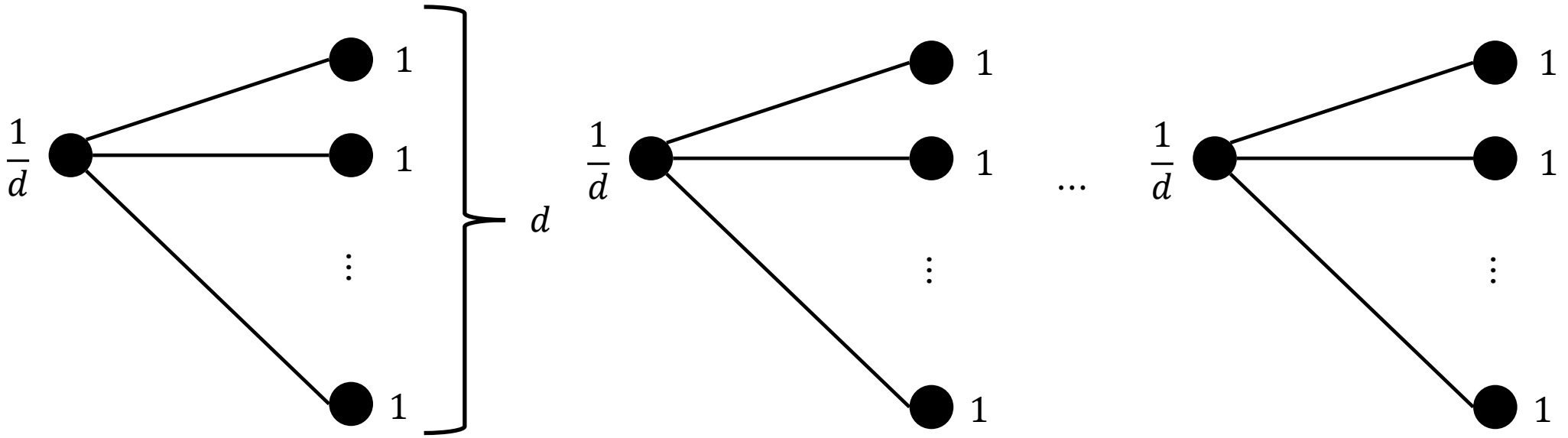
Question: Can we consider a concrete correlation model and design efficiently-computable, provably near-optimal policies?

- **Non-adaptive algorithms:** Fix up-front all edge probes
  - $O(max\ degree)$-approximation
  - Adaptivity gap is $\Omega(max\ degree)$
- **Adaptive algorithms:**
  - $O(density)$-approximation

All can be extended to knapsack case
(losing additional factors depending on edge costs)

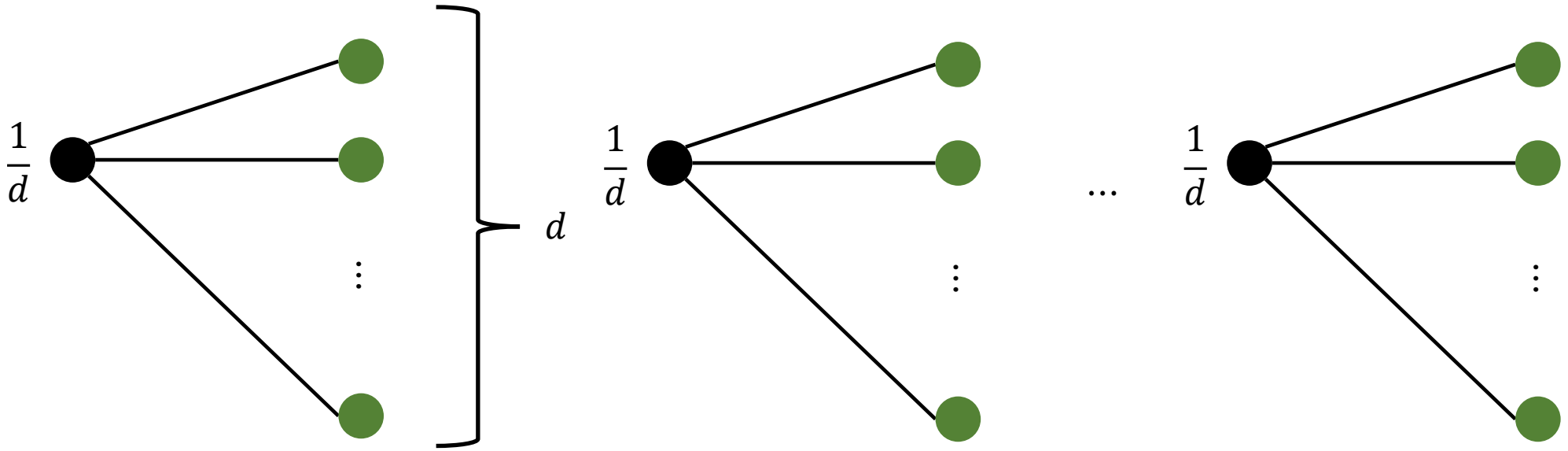
Example

- d stars with d leaves each
- Budget $k = d$



Example

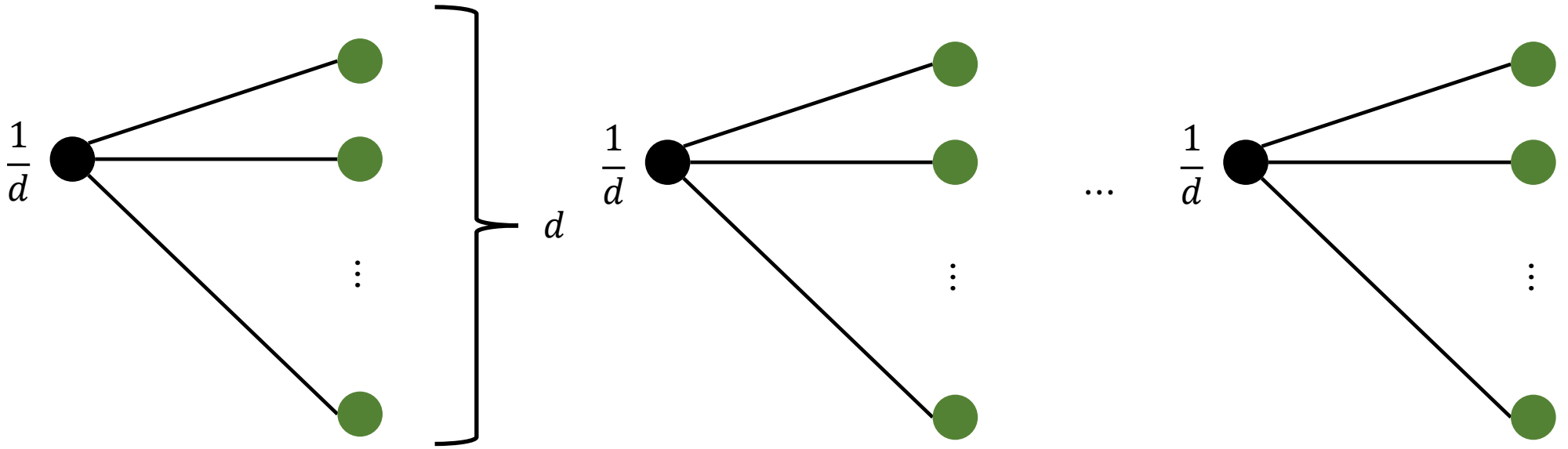
- d stars with d leaves each
- Budget $k = d$



Example

Non adaptive: Probe d arbitrary edges
$\Rightarrow \mathbb{E}\left[Non - adapt\right] = d \cdot \frac{1}{d} = 1$

- d stars with d leaves each
- Budget $k = d$

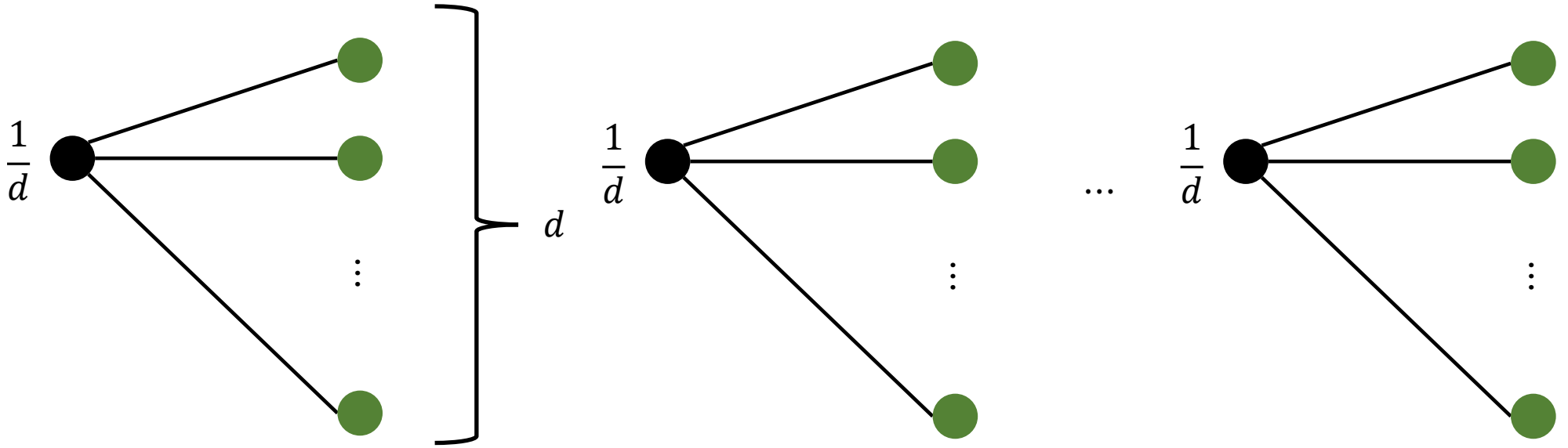


Example

- d stars with d leaves each
- Budget $k = d$

Non adaptive: Probe d arbitrary edges
$\Rightarrow \mathbb{E}\left[Non - adapt\right] = d \cdot \frac{1}{d} = 1$

Adaptive: Probe 1 edge per star until we find an active one; then probe rest of that star
$\Rightarrow \mathbb{E}\left[Adapt\right] = \Omega(d)$

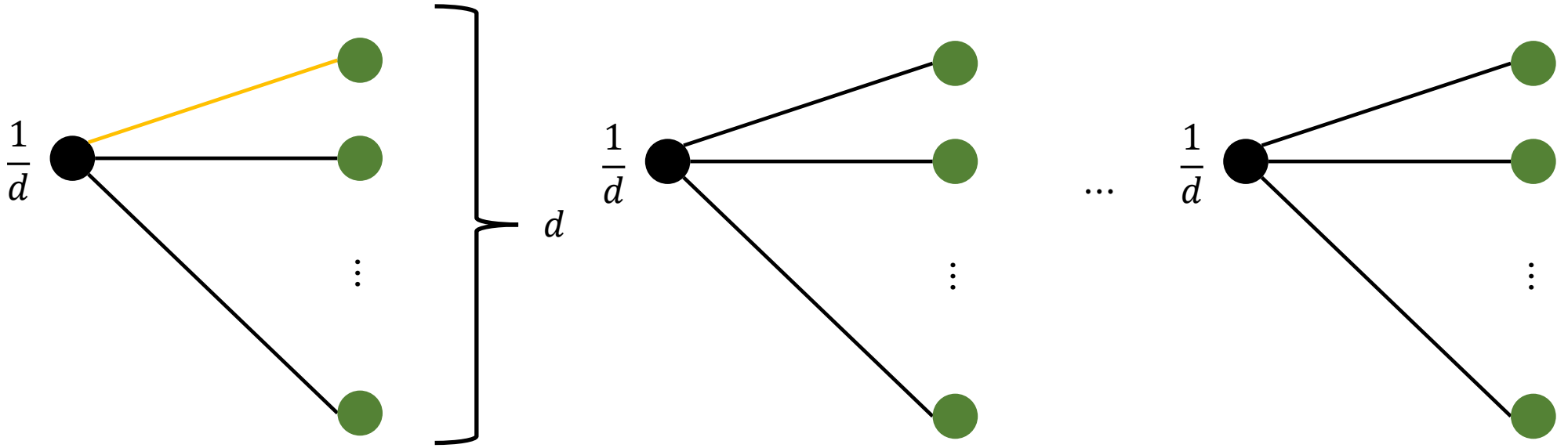


Example

- d stars with d leaves each
- Budget $k = d$

Non adaptive: Probe d arbitrary edges
$\Rightarrow \mathbb{E}\left[Non - adapt\right] = d \cdot \frac{1}{d} = 1$

Adaptive: Probe 1 edge per star until we find an active one; then probe rest of that star
$\Rightarrow \mathbb{E}\left[Adapt\right] = \Omega(d)$

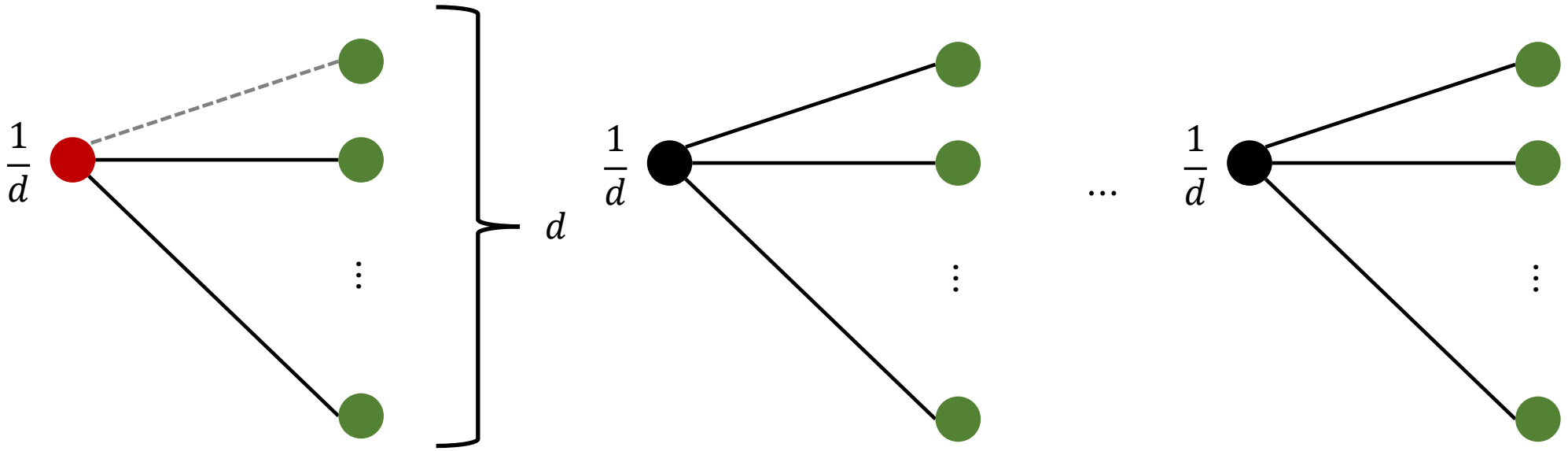


Example

- d stars with d leaves each
- Budget $k = d$

Non adaptive: Probe d arbitrary edges
$\Rightarrow \mathbb{E}\left[Non - adapt\right] = d \cdot \frac{1}{d} = 1$

Adaptive: Probe 1 edge per star until we find an active one; then probe rest of that star
$\Rightarrow \mathbb{E}\left[Adapt\right] = \Omega(d)$

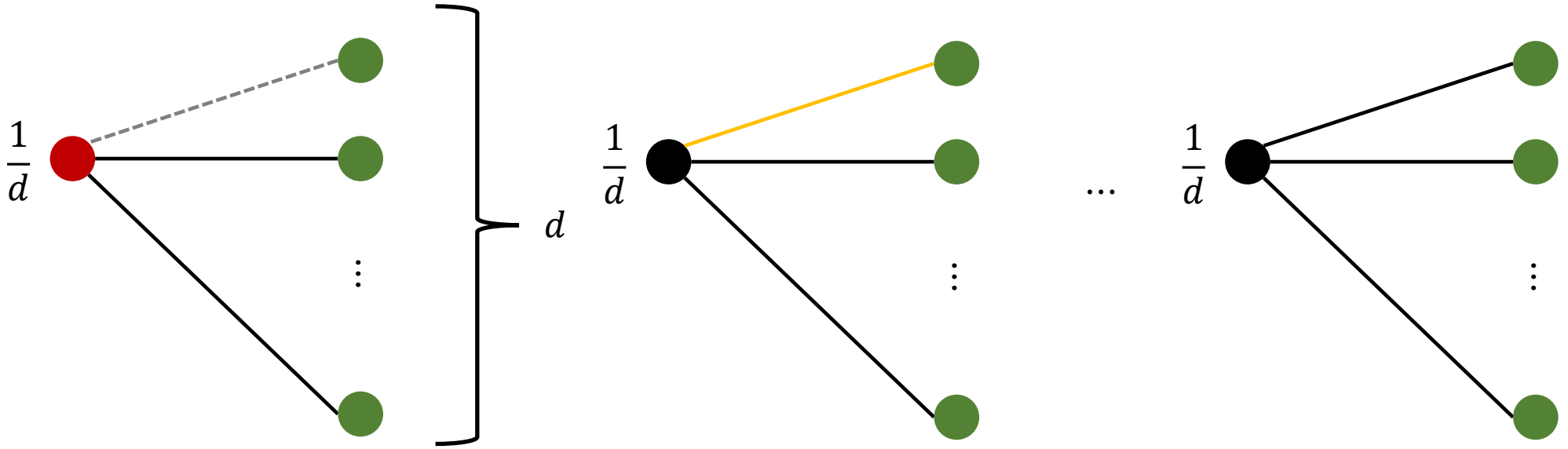


Example

- d stars with d leaves each
- Budget $k = d$

Non adaptive: Probe d arbitrary edges
$\Rightarrow \mathbb{E}\left[Non - adapt\right] = d \cdot \frac{1}{d} = 1$

Adaptive: Probe 1 edge per star until we find an active one; then probe rest of that star
$\Rightarrow \mathbb{E}\left[Adapt\right] = \Omega(d)$

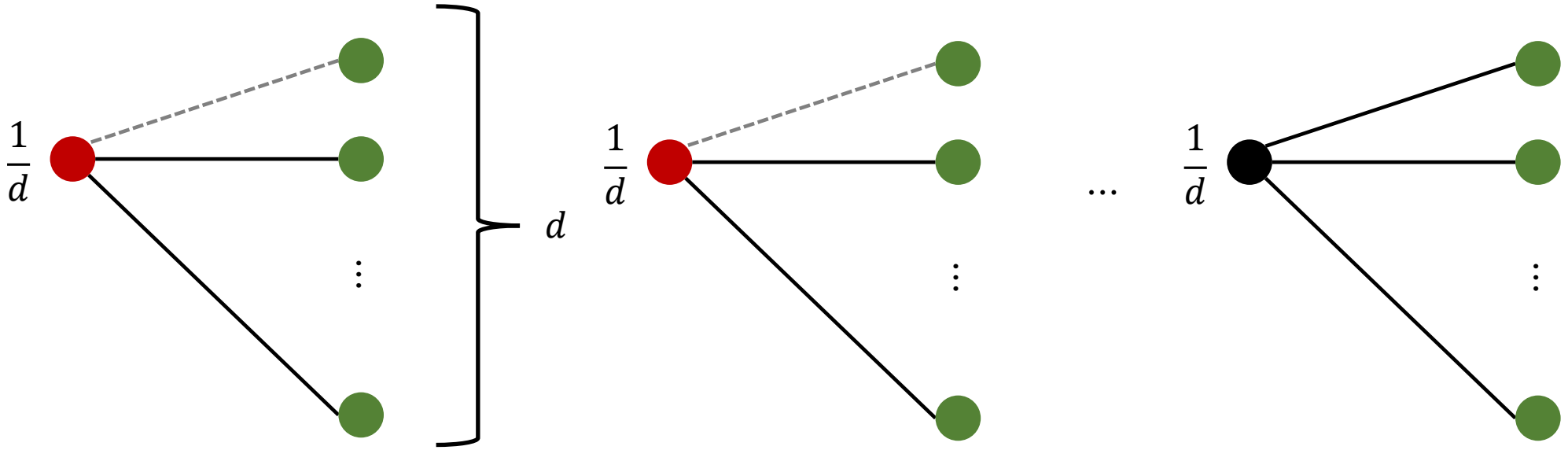


Example

- d stars with d leaves each
- Budget $k = d$

Non adaptive: Probe d arbitrary edges
$\Rightarrow \mathbb{E}\left[Non - adapt\right] = d \cdot \frac{1}{d} = 1$

Adaptive: Probe 1 edge per star until we find an active one; then probe rest of that star
$\Rightarrow \mathbb{E}\left[Adapt\right] = \Omega(d)$

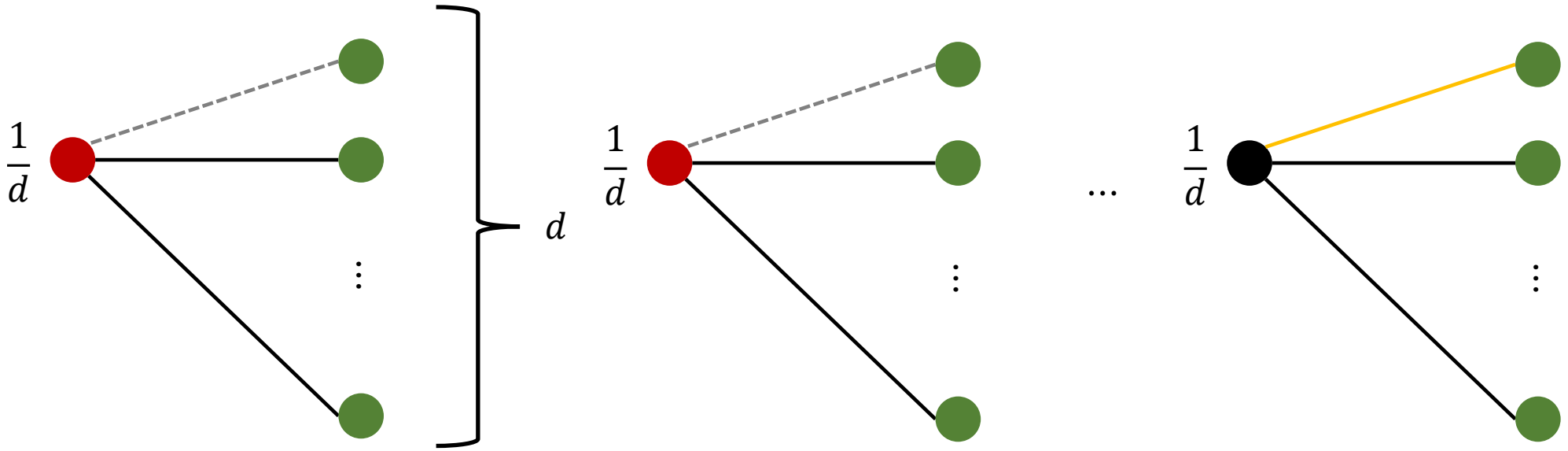


Example

- d stars with d leaves each
- Budget $k = d$

Non adaptive: Probe d arbitrary edges
$\Rightarrow \mathbb{E}\,[Non - adapt] = d \cdot \frac{1}{d} = 1$

Adaptive: Probe 1 edge per star until we find an active one; then probe rest of that star
$\Rightarrow \mathbb{E}\,[Adapt] = \Omega(d)$

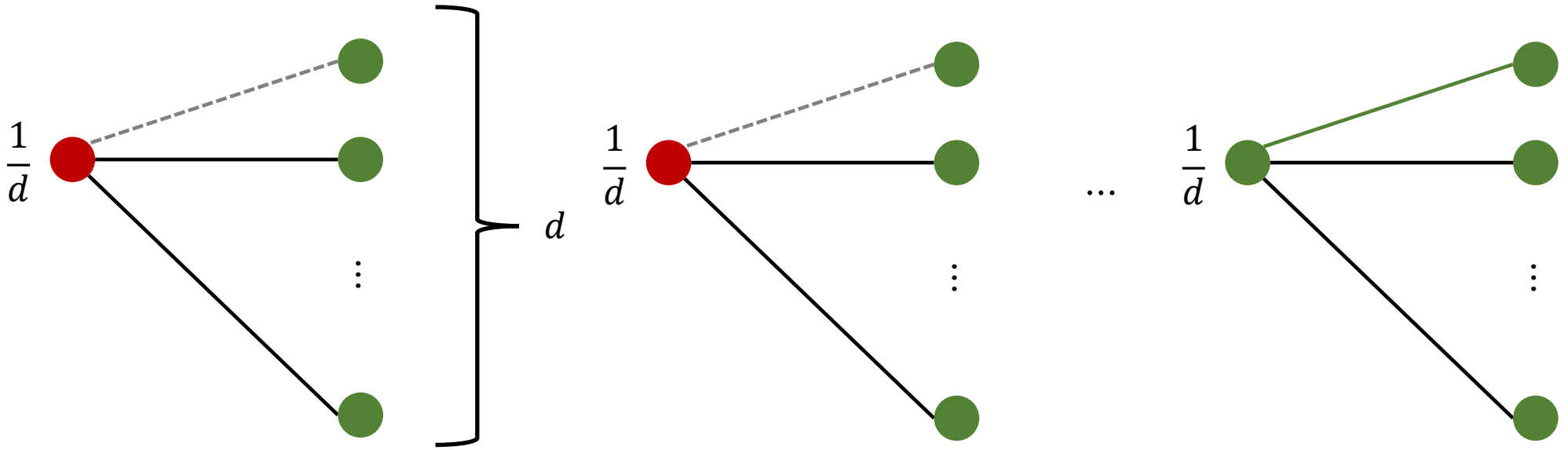


Example

- d stars with d leaves each
- Budget $k = d$

Non adaptive: Probe d arbitrary edges
$\Rightarrow \mathbb{E}\left[Non - adapt\right] = d \cdot \frac{1}{d} = 1$

Adaptive: Probe 1 edge per star until we find an active one; then probe rest of that star
$\Rightarrow \mathbb{E}\left[Adapt\right] = \Omega(d)$



Non-Adaptive Algorithms

- **Trivial Algorithm:** Pick k edges with largest $\Pr(e\ active)$

Non-Adaptive Algorithms

- **Trivial Algorithm:** Pick k edges with largest $\Pr(e\ active)$
- **Problem:** Natural knapsack LP is not a good benchmark

$$\max_{x} \sum_{e \in E} \Pr(e\ active) \cdot x_e$$

$$s.t. \sum_{e \in E} x_e \leq k$$

$$0 \leq x \leq 1$$

Non-Adaptive Algorithms

- **Trivial Algorithm:** Pick k edges with largest $\Pr(e\ active)$
- **Problem:** Natural knapsack LP is not a good benchmark

$$\max_{x} \sum_{e \in E} \Pr(e\ active) \cdot x_e$$

$$s.t. \sum_{e \in E} x_e \leq k$$

$$0 \leq x \leq 1$$

Standard Analysis:

- Let x's be marginals of optimal adaptive policy: $x_e = \Pr(OPT\ probes\ e)$

Non-Adaptive Algorithms

- **Trivial Algorithm:** Pick k edges with largest $\Pr(e\ active)$
- **Problem:** Natural knapsack LP is not a good benchmark

$$\max_{x} \sum_{e \in E} \Pr(e\ active) \cdot x_e$$

$$s.t. \sum_{e \in E} x_e \leq k$$

$$0 \leq x \leq 1$$

Standard Analysis:

- Let x's be marginals of optimal adaptive policy: $x_e = \Pr(OPT\ probes\ e)$
- Compute objective value:

$$\sum_{e \in E} \Pr(e\ active) \cdot x_e = \sum_{e \in E} \Pr(e\ active) \cdot \Pr(OPT\ probes\ e)$$

Non-Adaptive Algorithms

- **Trivial Algorithm:** Pick k edges with largest $\Pr(e\ active)$
- **Problem:** Natural knapsack LP is not a good benchmark

$$\max_x \sum_{e \in E} \Pr(e\ active) \cdot x_e$$

$$s.t. \sum_{e \in E} x_e \leq k$$

$$0 \leq x \leq 1$$

Standard Analysis:

- Let x's be marginals of optimal adaptive policy: $x_e = \Pr(OPT\ probes\ e)$
- Compute objective value:

$$\sum_{e \in E} \Pr(e\ active) \cdot x_e = \sum_{e \in E} \Pr(e\ active) \cdot \Pr(OPT\ probes\ e)$$

$$= \sum_{e \in E} \Pr(e\ active \wedge OPT\ probes\ e)$$

$$= \mathbb{E}[OPT]$$

Non-Adaptive Algorithms

- **Trivial Algorithm:** Pick k edges with largest $\Pr(e\ active)$
- **Problem:** Natural knapsack LP is not a good benchmark

$$\max_x \sum_{e \in E} \Pr(e\ active) \cdot x_e$$

$$s.t. \sum_{e \in E} x_e \leq k$$

$$0 \leq x \leq 1$$

Standard Analysis:

- Let x's be marginals of optimal adaptive policy: $x_e = \Pr(OPT\ probes\ e)$
- Compute objective value:

$$\sum_{e \in E} \Pr(e\ active) \cdot x_e = \sum_{e \in E} \Pr(e\ active) \cdot \Pr(OPT\ probes\ e)$$

$$= \sum_{e \in E} \Pr(e\ active \wedge OPT\ probes\ e)$$

OPT's decisions no longer independent of outcomes

Decomposition Lemma

Lemma: Let G be any graph and k any budget, where $G = \bigcup_\ell G_\ell$.
Then $\mathbb{E}[OPT(G,k)] \leq \sum_\ell \mathbb{E}[OPT(G_\ell, 2k)]$

Decomposition Lemma

Lemma: Let G be any graph and k any budget, where $G = \bigcup_\ell G_\ell$. Then $\mathbb{E}[OPT(G,k)] \leq \sum_\ell \mathbb{E}[OPT(G_\ell, 2k)]$

- **Proof Sketch:** Simulate OPT on subgraph G_ℓ

Decomposition Lemma

Lemma: Let G be any graph and k any budget, where $G = \bigcup_\ell G_\ell$. Then $\mathbb{E}[OPT(G,k)] \leq \sum_\ell \mathbb{E}[OPT(G_\ell, 2k)]$

- **Proof Sketch:** Simulate OPT on subgraph G_ℓ
- Decompose G into $O(\max degree)$-many matchings

Decomposition Lemma

Lemma: Let G be any graph and k any budget, where $G = \bigcup_\ell G_\ell$.
Then $\mathbb{E}[OPT(G,k)] \leq \sum_\ell \mathbb{E}[OPT(G_\ell, 2k)]$

- **Proof Sketch:** Simulate OPT on subgraph G_ℓ
- Decompose G into $O(\max degree)$-many matchings
- $\Rightarrow$ LP is now good benchmark on the best matching M

Decomposition Lemma

Lemma: Let G be any graph and k any budget, where $G = \bigcup_\ell G_\ell$.
Then $\mathbb{E}[OPT(G,k)] \leq \sum_\ell \mathbb{E}[OPT(G_\ell, 2k)]$

- **Proof Sketch:** Simulate OPT on subgraph G_ℓ
- Decompose G into $O(\max degree)$-many matchings
- $\Rightarrow$ LP is now good benchmark on the best matching M
- $\Rightarrow E[Alg] = LP(G,k) \geq LP(M,k)$

$$= \mathbb{E}[OPT(M,k)] \geq^* \frac{1}{\max degree} \cdot \mathbb{E}[OPT(G,k)]$$

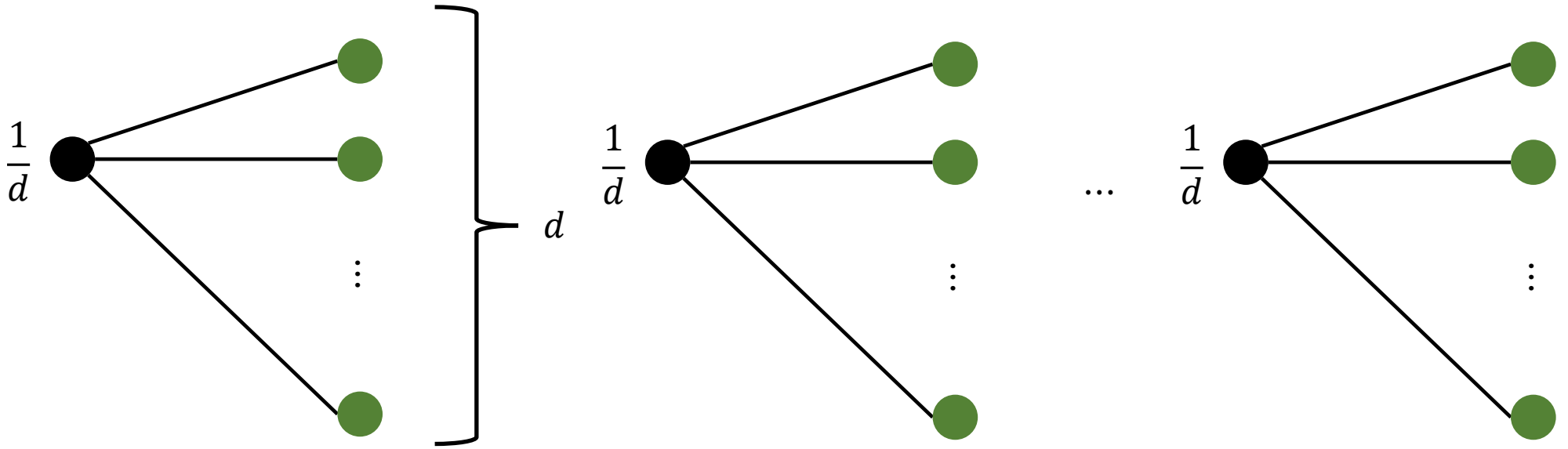
* Ignore budget scaling

Example

- d stars with d leaves each
- Budget $k = d$

Two Phase:

1. **(Explore)** Probe arbitrary edge of $\frac{d}{2}$ stars
2. **(Exploit)** Conditioned on 1. – non-adaptively probe the best $\frac{d}{2}$ edges

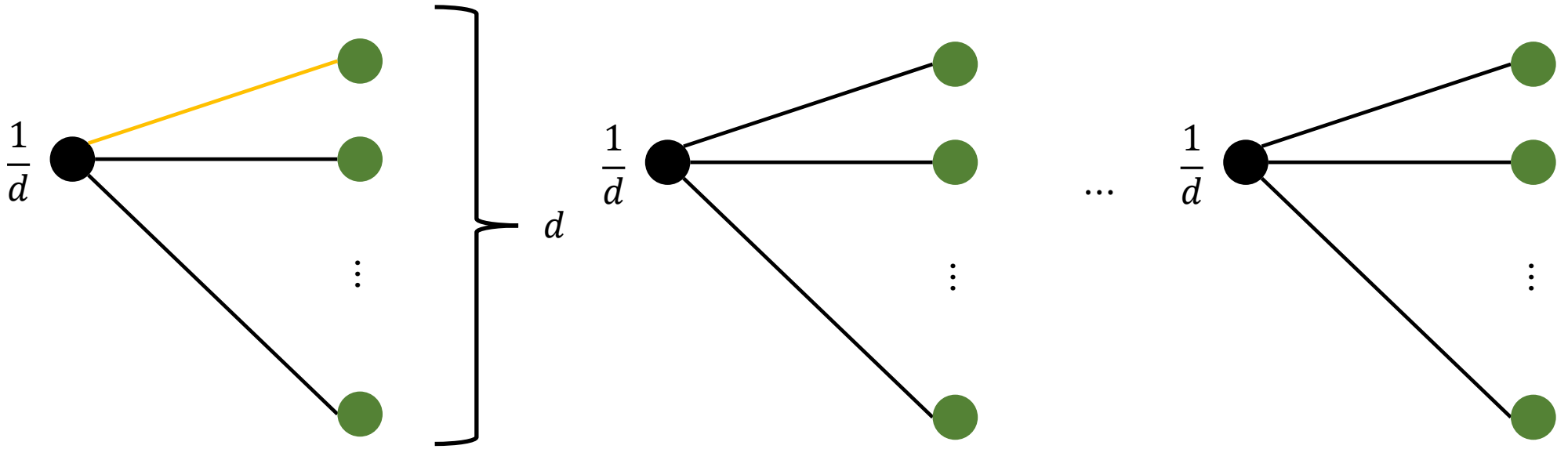


Example

- d stars with d leaves each
- Budget $k = d$

Two Phase:

1. **(Explore)** Probe arbitrary edge of $\frac{d}{2}$ stars
2. **(Exploit)** Conditioned on 1. – non-adaptively probe the best $\frac{d}{2}$ edges

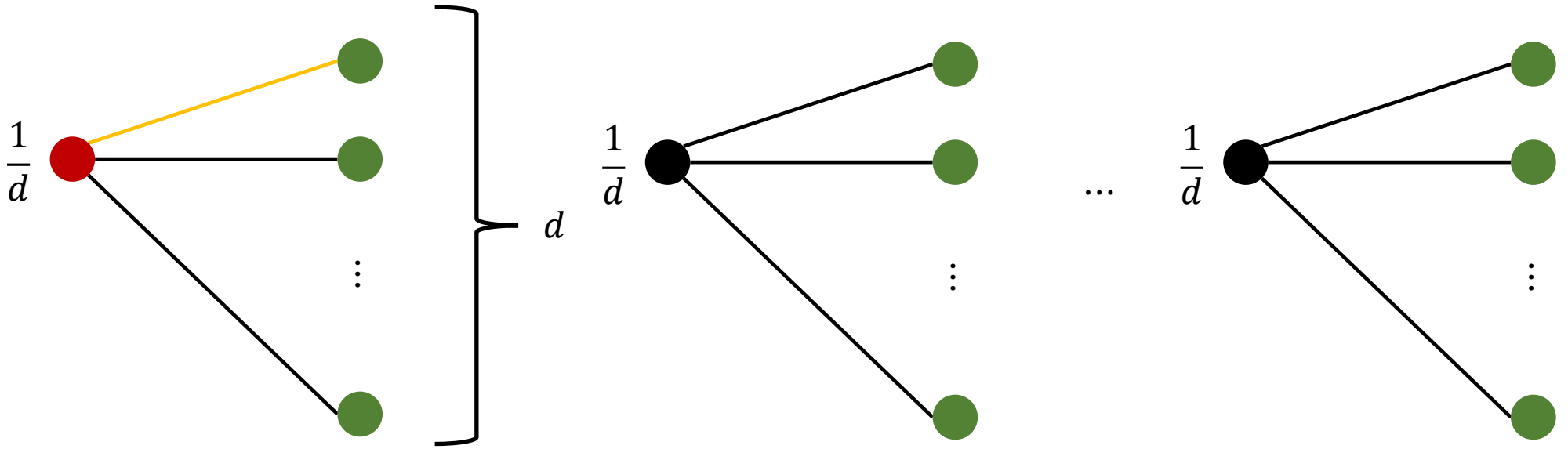


Example

- d stars with d leaves each
- Budget $k = d$

Two Phase:

1. **(Explore)** Probe arbitrary edge of $\frac{d}{2}$ stars
2. **(Exploit)** Conditioned on 1. – non-adaptively probe the best $\frac{d}{2}$ edges

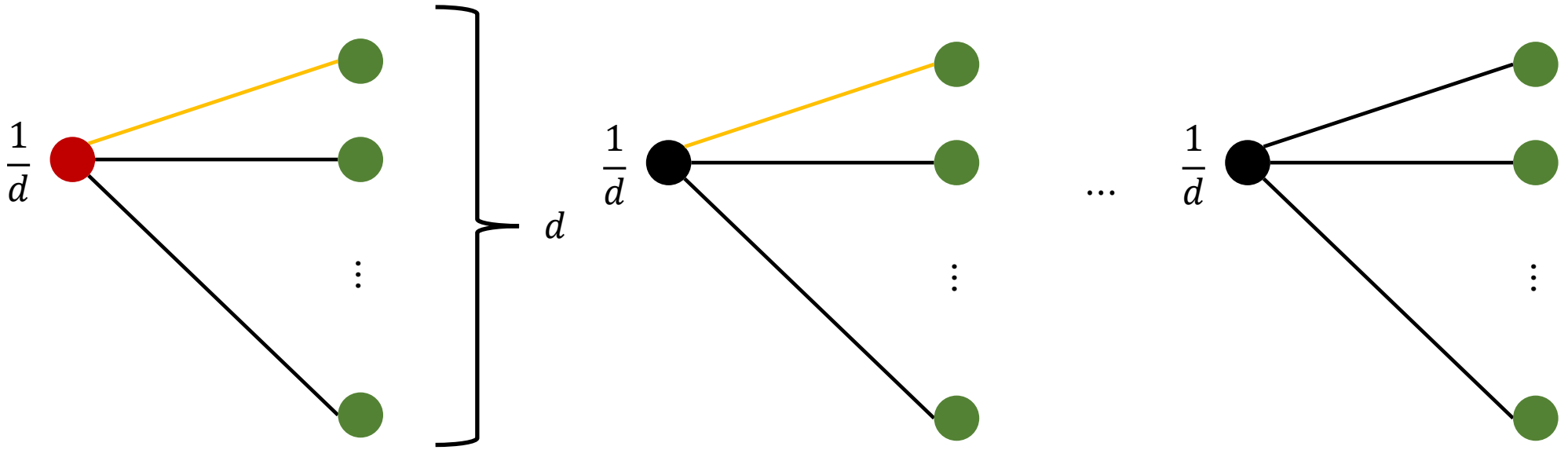


Example

- d stars with d leaves each
- Budget $k = d$

Two Phase:

1. **(Explore)** Probe arbitrary edge of $\frac{d}{2}$ stars
2. **(Exploit)** Conditioned on 1. – non-adaptively probe the best $\frac{d}{2}$ edges

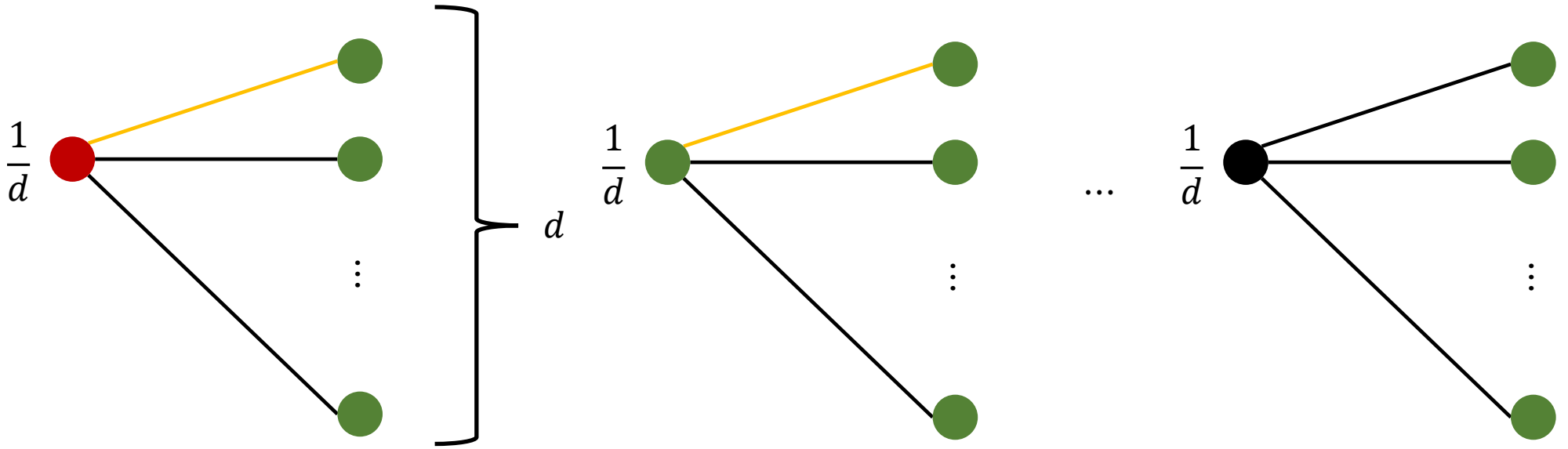


Example

- d stars with d leaves each
- Budget $k = d$

Two Phase:

1. **(Explore)** Probe arbitrary edge of $\frac{d}{2}$ stars
2. **(Exploit)** Conditioned on 1. – non-adaptively probe the best $\frac{d}{2}$ edges

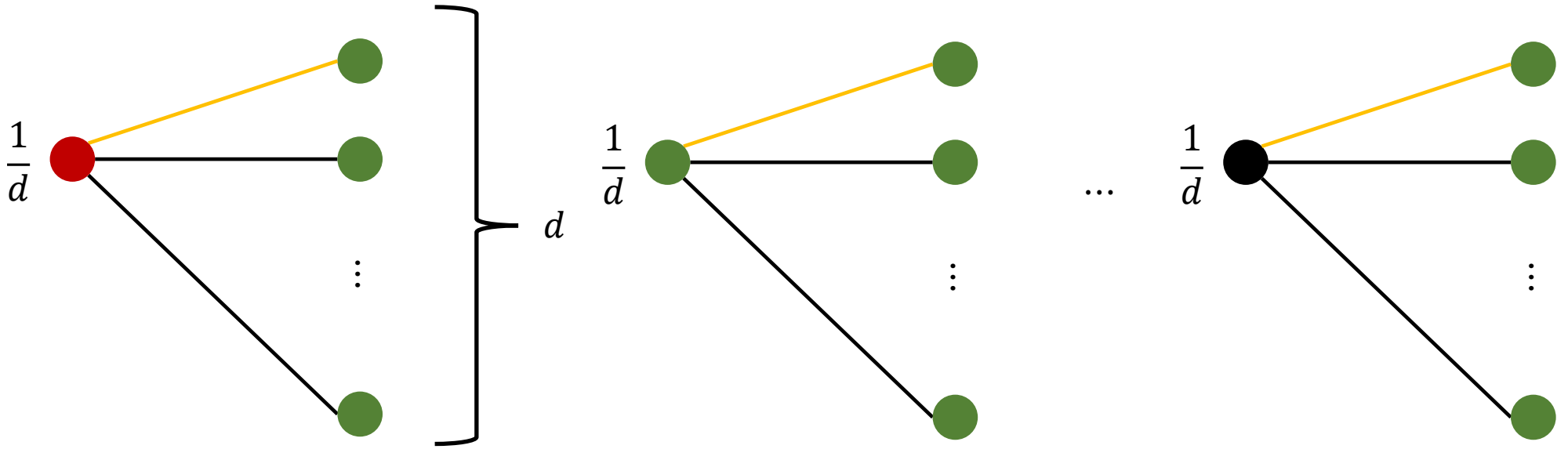


Example

- d stars with d leaves each
- Budget $k = d$

Two Phase:

1. **(Explore)** Probe arbitrary edge of $\frac{d}{2}$ stars
2. **(Exploit)** Conditioned on 1. – non-adaptively probe the best $\frac{d}{2}$ edges

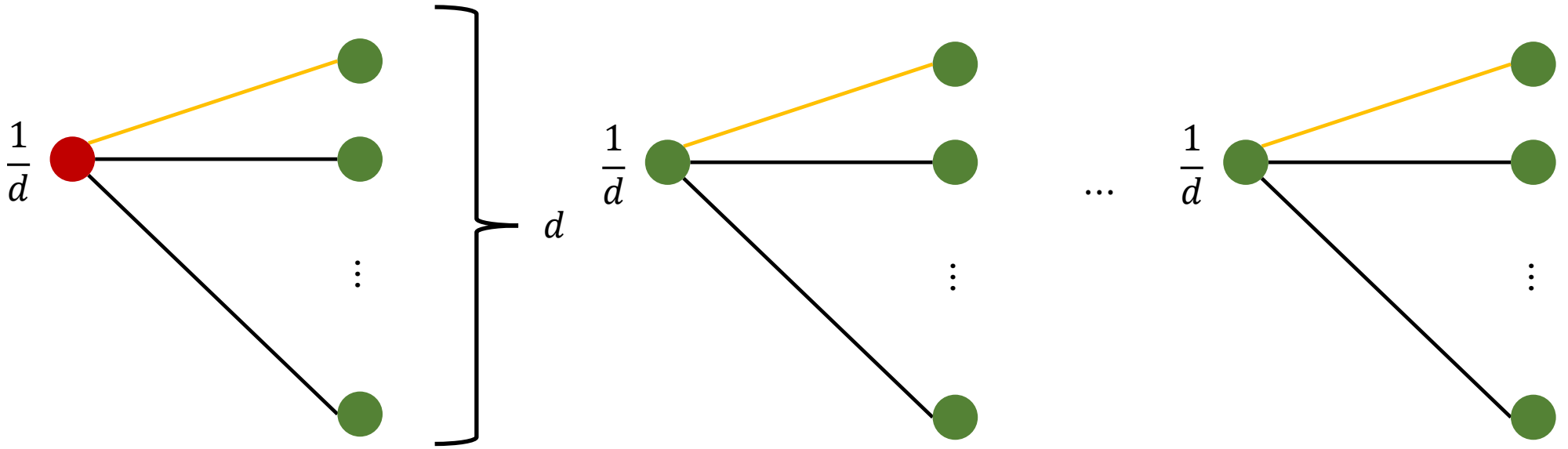


Example

- d stars with d leaves each
- Budget $k = d$

Two Phase:

1. **(Explore)** Probe arbitrary edge of $\frac{d}{2}$ stars
2. **(Exploit)** Conditioned on 1. – non-adaptively probe the best $\frac{d}{2}$ edges

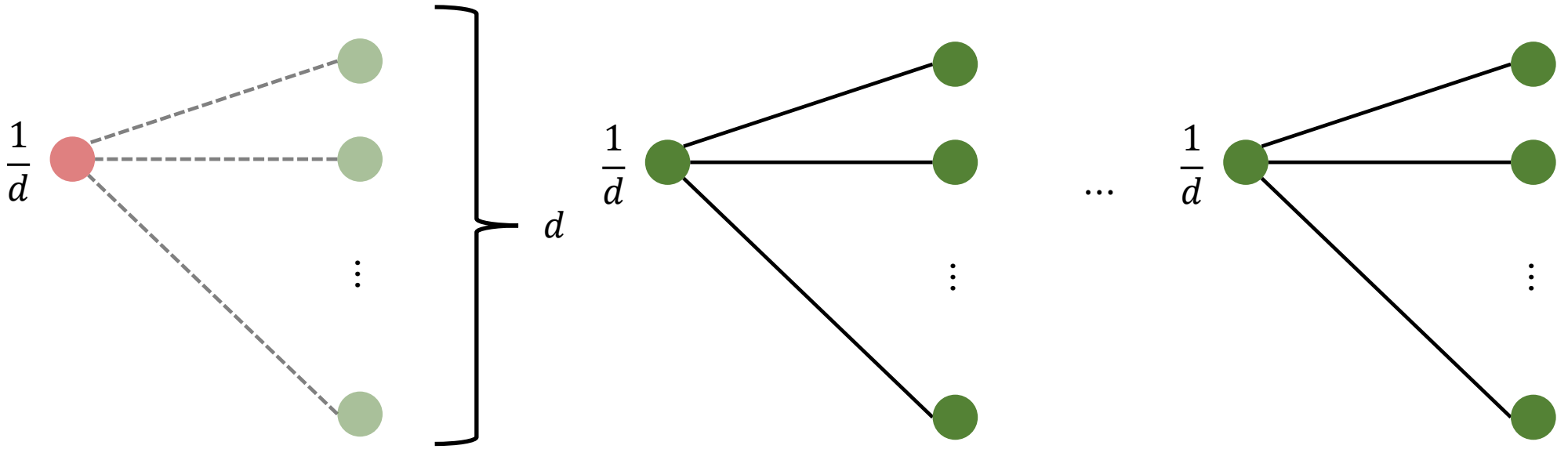


Example

- d stars with d leaves each
- Budget $k = d$

Two Phase:

1. **(Explore)** Probe arbitrary edge of $\frac{d}{2}$ stars
2. **(Exploit)** Conditioned on 1. – non-adaptively probe the best $\frac{d}{2}$ edges

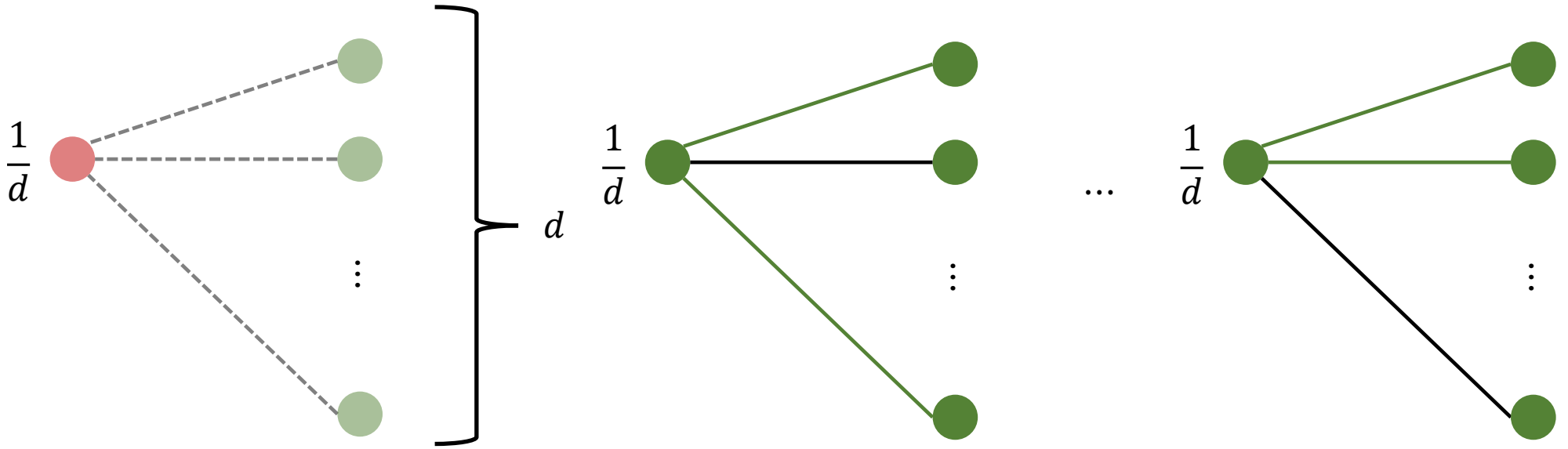


Example

- d stars with d leaves each
- Budget $k = d$

Two Phase:

1. **(Explore)** Probe arbitrary edge of $\frac{d}{2}$ stars
2. **(Exploit)** Conditioned on 1. – non-adaptively probe the best $\frac{d}{2}$ edges



Two Phase Algorithm

- Use Decomposition Lemma to reduce arbitrary G to collection of stars (lose $density$ by averaging)

Two Phase Algorithm

- Use Decomposition Lemma to reduce arbitrary G to collection of stars
  (lose $density$ by averaging)

Lemma: $O(1)$-approximation for collection of stars

Two Phase Algorithm

- Use Decomposition Lemma to reduce arbitrary G to collection of stars (lose $density$ by averaging)

Lemma: $O(1)$-approximation for collection of stars

- **Explore**
  - Think of first $\frac{k}{2}$ queries as vertex probes (learn single vertex value; ignore other endpoint)
  - Greedy algorithm to maximize conditional reward obtained by final $\frac{k}{2}$ queries

Two Phase Algorithm

- Use Decomposition Lemma to reduce arbitrary G to collection of stars (lose $density$ by averaging)

Lemma: $O(1)$-approximation for collection of stars

- **Explore**
  - Think of first $\frac{k}{2}$ queries as vertex probes (learn single vertex value; ignore other endpoint)
  - Greedy algorithm to maximize conditional reward obtained by final $\frac{k}{2}$ queries
- **Exploit**
  - Conditioned on exploration, pick best $\frac{k}{2}$ edges

Our Results

Question: Can we consider a concrete correlation model and design efficiently-computable, provably near-optimal policies?

- **Non-adaptive algorithms:** Fix up-front all edge probes
  - $O(max\ degree)$-approximation
  - Adaptivity gap is $\Omega(max\ degree)$
- **Adaptive algorithms:**
  - $O(density)$-approximation

All can be extended to knapsack case
(losing additional factors depending on edge costs)

Summary

Question: Can we consider a concrete correlation model and design efficiently-computable, provably near-optimal policies?

- Introduce Bayesian Probing on graphs
- **Open Questions:**
  - Hypergraph
  - More general vertex distributions
  - More general edge reward functions
  - Different information from probes