On-Line Social Systems with Long-Range Goals

Jon Kleinberg

Cornell University

Including joint work with Ashton Anderson, Dan Huttenlocher, Jure Leskovec, and Sigal Oren.

Long-Range Planning

Growth in on-line systems where users have long visible lifetimes and set long-range goals.

- Reputation, promotion, status, individual achievement.

How should we model individual decision-making in these settings with long-range planning?

Badges

Structural framework for analysis: state space of activities.

- User lifetimes correspond to trajectories through state space.
- Effort incurs cost, leads to rewards.

On-line domain: badges and related incentives as reward systems.

- Social-psychological dimensions [Antin-Churchill 2011]
- Game-theoretic [Deterding et al 2011, Easley-Ghosh 2013]
- Contest/auction-based [Cavallo-Jain 12, Chawla-Hartline-Sivan 12]

Outline

Model the interaction of incentives and long-range planning in state spaces representing actions on site.

(1) Cumulative rewards: milestones for effort
[Anderson-Huttenlocher-Kleinberg-Leskovec]

- A basic model of an individual working toward long-range rewards.
- Exploration of the model on StackOverflow
- Experiments with MOOC forums on Coursera

Outline

Model the interaction of incentives and long-range planning in state spaces representing actions on site.

(1) Cumulative rewards: milestones for effort
[Anderson-Huttenlocher-Kleinberg-Leskovec]

- A basic model of an individual working toward long-range rewards.
- Exploration of the model on StackOverflow
- Experiments with MOOC forums on Coursera

(2) Incentives and planning with time-inconsistent behavior
[Kleinberg-Oren]

- Start from principles in behavioral economics
[Strotz 1955, Pollak 1968, Akerlof 1991, Laibson 1997]
- Develop a graph-theoretic model to represent planning as path-finding with a behavioral bias.

First Domain for Analysis: Stack Overflow

stackoverflow Questions Tags Users Badges Unanswered

Connected components in a graph with 100 million nodes

1

I am trying to get the list of connected components in a graph with 100 million nodes. For smaller graphs, I usually use the connected_components function of the Networkx module in Python which does exactly that. However, loading a graph with 100 million nodes (and their edges) into memory with this module would require ca. 110GB of memory, which I don't have. An alternative would be to use a graph database which has a connected components function but I haven't found any in Python. It would seem that Dex (API: Java, .NET, C++) has this functionality but I'm not 100% sure. Ideally I'm looking for a solution in Python. Many thanks.

python graph

share | improve this question

asked Jun 13 '12 at 13:48
user1453508
27 •4

1 Answer

active oldest votes

1

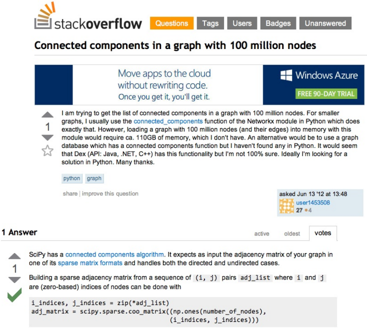
SciPy has a connected components algorithm. It expects as input the adjacency matrix of your graph in one of its sparse matrix formats and handles both the directed and undirected cases.

Building a sparse adjacency matrix from a sequence of `(i, j)` pairs `adj_list` where `i` and `j` are (zero-based) indices of nodes can be done with

Basic Model

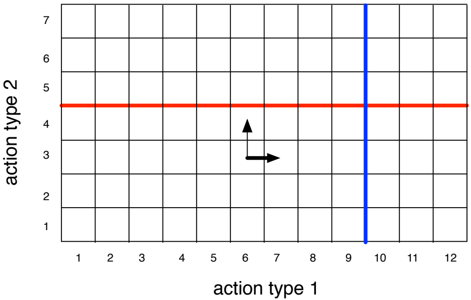
A population of users and a site designer.

- Designer wants certain frequency of activites.
- Designer creates badges, which have value to users.



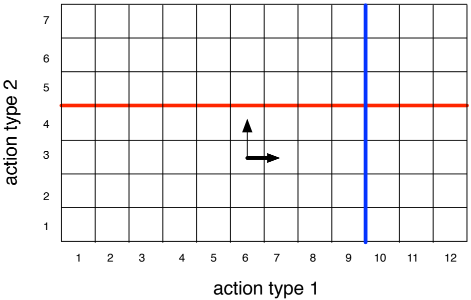
Our Model

- Action types $A_1, A_2, \ldots, A_n$. (ask, answer, vote, *off-site*, ...)
- User's state is n-dimensional.
- User has preferred distribution **p** over action types.
- User exits system with probability $\delta > 0$ each step.



Our Model

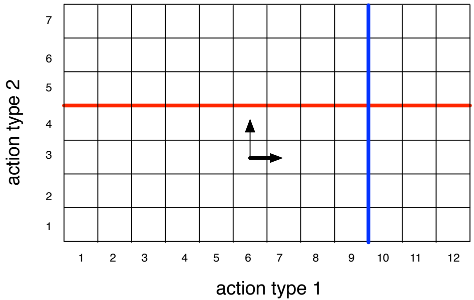
- Action types $A_1, A_2, \ldots, A_n$.
  (ask, answer, vote, *off-site*, ...)
- User's state is n-dimensional.
- User has preferred distribution $\mathbf{p}$ over action types.
- User exits system with probability $\delta > 0$ each step.



- Each badge b is a monotone subset of the state space; reward V_b is conferred when the user enters this subset.

Our Model

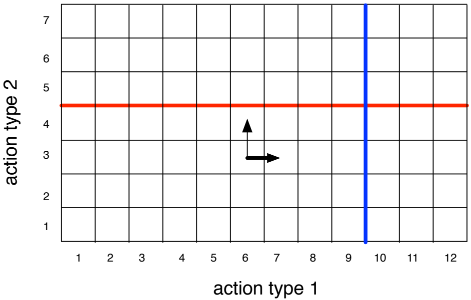
- Action types $A_1, A_2, \ldots, A_n$.
  (ask, answer, vote, *off-site*, ...)
- User's state is n-dimensional.
- User has preferred distribution $\mathbf{p}$ over action types.
- User exits system with probability $\delta > 0$ each step.



- Each badge b is a monotone subset of the state space; reward V_b is conferred when the user enters this subset.
- User can pick distribution $\mathbf{x} \neq \mathbf{p}$ to get badge more quickly; comes at a cost $g(\mathbf{x}, \mathbf{p})$.

Our Model

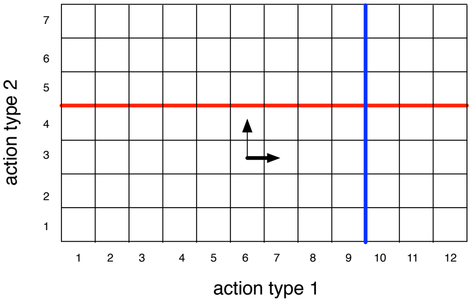
- Action types $A_1, A_2, \ldots, A_n$.
  (ask, answer, vote, *off-site*, ...)
- User's state is n-dimensional.
- User has preferred distribution $\mathbf{p}$ over action types.
- User exits system with probability $\delta > 0$ each step.



- Each badge b is a monotone subset of the state space; reward V_b is conferred when the user enters this subset.
- User can pick distribution $\mathbf{x} \neq \mathbf{p}$ to get badge more quickly; comes at a cost $g(\mathbf{x}, \mathbf{p})$.
- User optimization: Choose $\mathbf{x_a} = (\mathbf{x}_\mathbf{a}^1, \ldots, \mathbf{x}_\mathbf{a}^n)$ in each state $\mathbf{a}$ to optimize utility $U(\mathbf{x_a})$.

Our Model

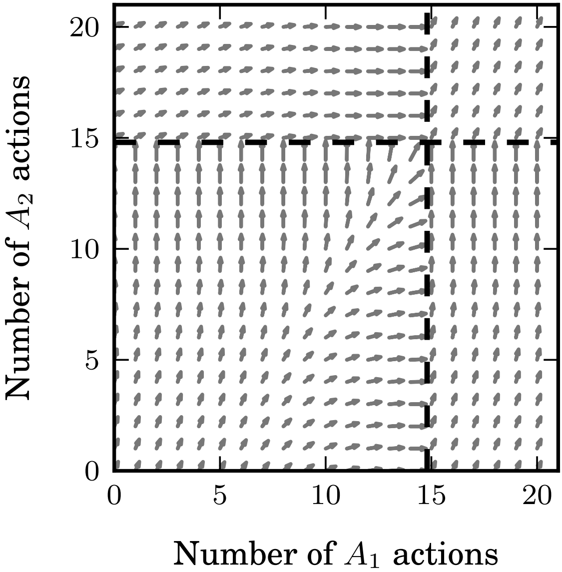
- Action types $A_1, A_2, \ldots, A_n$.
  (ask, answer, vote, *off-site*, ...)
- User's state is n-dimensional.
- User has preferred distribution $\mathbf{p}$ over action types.
- User exits system with probability $\delta > 0$ each step.



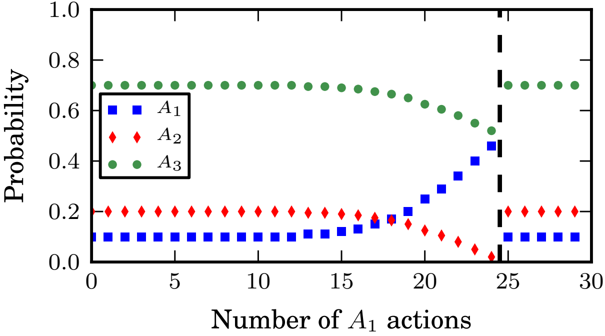
- Each badge b is a monotone subset of the state space; reward V_b is conferred when the user enters this subset.
- User can pick distribution $\mathbf{x} \neq \mathbf{p}$ to get badge more quickly; comes at a cost $g(\mathbf{x}, \mathbf{p})$.
- User optimization: Choose $\mathbf{x_a} = (\mathbf{x}^1_\mathbf{a}, \ldots, \mathbf{x}^n_\mathbf{a})$ in each state $\mathbf{a}$ to optimize utility $U(\mathbf{x_a})$.

$$U(\mathbf{x_a}) = \sum_{b \text{ won}} V_b - g(\mathbf{x_a}, \mathbf{p}) + (1-\delta) \sum_{i=1}^{n} \mathbf{x}^i_\mathbf{a} \cdot U(\mathbf{x}_{\mathbf{a}+\mathbf{e_i}})$$

What a Solution Looks Like



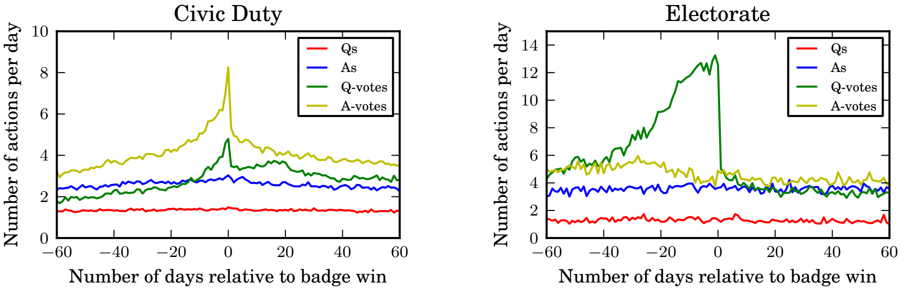
A One-Dimensional Version



Example: Badge at 25 actions of type 1.

- Canonical behavior: user "steers" in A_1 direction; then resets after receiving the badge.

Evaluating Qualitative Predictions

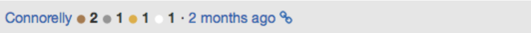


Questions related to badge-based incentives:

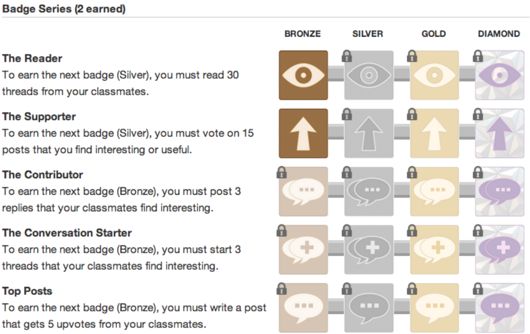
- The Badge Placement Problem:
  Given a desired mixture of actions, how should one define badges to (approximately) induce these actions?
- How do badges derive their value?
  Social / Motivational / Transactional?

An Experiment on Coursera

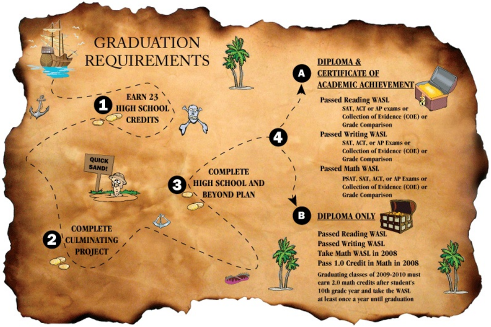
Thread byline:



Badge ladder:



Planning and Time-Inconsistency



Tacoma Public School System

Our models thus far:

- Plans are multi-step.
- Agents chooses optimal sequence given costs and benefits.

What could go wrong?

- Costs and benefits are unknown, and/or genuinely changing over time.
- Time-inconsistency.

Planning and Time-Inconsistency

Our models thus far:

- Plans are multi-step.
- Agents chooses optimal sequence given costs and benefits.

What could go wrong?

- Costs and benefits are unknown, and/or genuinely changing over time.
- Time-inconsistency.

Why did George Akerlof not make it to the post office?

Agent must ship a package sometime in next n days.

- One-time effort cost c to ship it.
- Loss-of-use cost x each day hasn't been shipped.

Why did George Akerlof not make it to the post office?

Agent must ship a package sometime in next n days.

- One-time effort cost c to ship it.
- Loss-of-use cost x each day hasn't been shipped.

An optimization problem:

- If shipped on day t, cost is $c + tx$.
- Goal: $\min_{1 \leq t \leq n} c + tx$.
- Optimized at $t = 1$.

Why did George Akerlof not make it to the post office?

Agent must ship a package sometime in next n days.

- One-time effort cost c to ship it.
- Loss-of-use cost x each day hasn't been shipped.

An optimization problem:

- If shipped on day t, cost is $c + tx$.
- Goal: $\min_{1 \leq t \leq n} c + tx$.
- Optimized at $t = 1$.

In Akerlof's story, he was the agent, and he *procrastinated*:

- Each day he planned that he'd do it tomorrow.
- Effect: waiting until day n, when it must be shipped, and doing it then, at a significantly higher cumulative cost.

Why did George Akerlof not make it to the post office?

Agent must ship a package sometime in next n days.

- One-time effort cost c to ship it.
- Loss-of-use cost x each day hasn't been shipped.

A model based on present bias [Akerlof 91; cf. Strotz 55, Pollak 68]

- Costs incurred today are more salient: raised by factor $b > 1$.

On day t:

- Remaining cost if sent today is bc.
- Remaining cost if sent tomorrow is $bx + c$.
- Tomorrow is preferable if $(b - 1)c > bx$.

Why did George Akerlof not make it to the post office?

Agent must ship a package sometime in next n days.

- One-time effort cost c to ship it.
- Loss-of-use cost x each day hasn't been shipped.

A model based on present bias [Akerlof 91; cf. Strotz 55, Pollak 68]

- Costs incurred today are more salient: raised by factor $b > 1$.

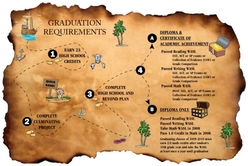
On day t:

- Remaining cost if sent today is bc.
- Remaining cost if sent tomorrow is $bx + c$.
- Tomorrow is preferable if $(b - 1)c > bx$.

General framework: quasi-hyperbolic discounting [Laibson 1997]

- Cost/reward c realized t units in future has present value $\beta\delta^t c$
- Special case: $\delta = 1$, $b = \beta^{-1}$, and agent is naive about bias.
- Can model procrastination, task abandonment [O'Donoghue-Rabin08], and benefits of choice reduction [Ariely and Wertenbroch 02, Kaur-Kremer-Mullainathan 10]

Cost Ratio



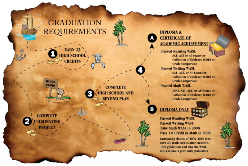
Cost ratio:

$$\frac{\text{Cost incurred by present-biased agent}}{\text{Minimum cost achievable}}$$

Across all stories in which present bias has an effect, what's the worst cost ratio?

$$\max_{\text{stories } S} \quad \text{cost ratio}(S).$$

Cost Ratio



Cost ratio:

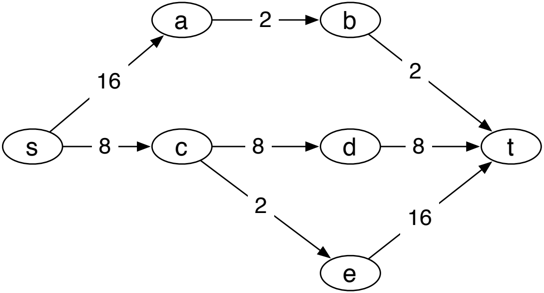
$$\frac{\text{Cost incurred by present-biased agent}}{\text{Minimum cost achievable}}$$

Across all stories in which present bias has an effect, what's the worst cost ratio?

$$\max_{\text{stories } S} \text{cost ratio}(S).$$

???

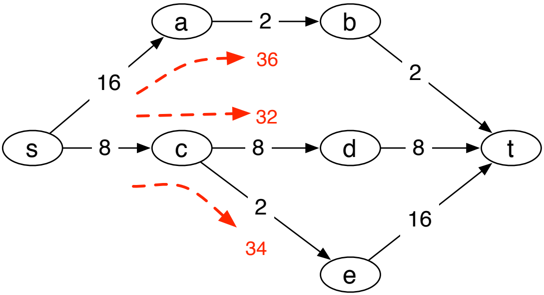
A Graph-Theoretic Framework



Use graphs as basic structure to represent scenarios.

- Agent plans to follow cheapest path from s to t.
- From a given node, immediately outgoing edges have costs multplied by $b > 1$.

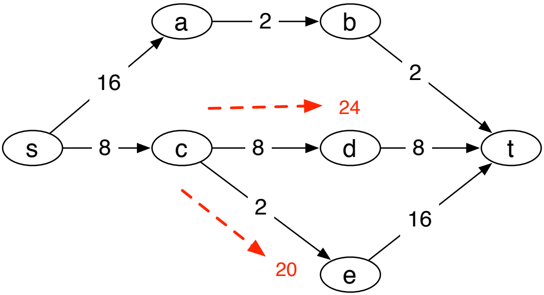
A Graph-Theoretic Framework



Use graphs as basic structure to represent scenarios.

- Agent plans to follow cheapest path from s to t.
- From a given node, immediately outgoing edges have costs multplied by $b > 1$.

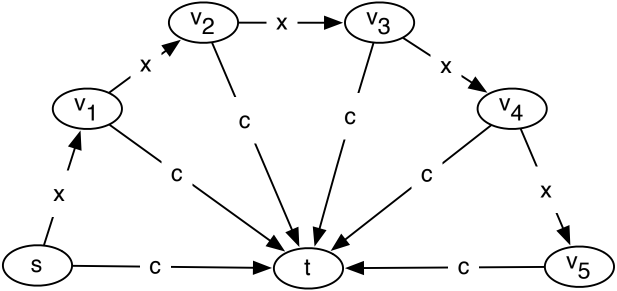
A Graph-Theoretic Framework



Use graphs as basic structure to represent scenarios.

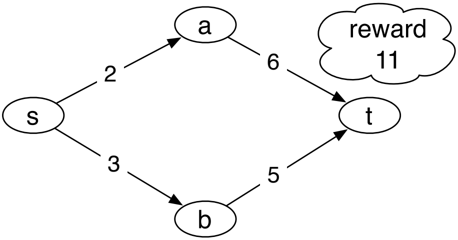
- Agent plans to follow cheapest path from s to t.
- From a given node, immediately outgoing edges have costs multplied by $b > 1$.

Example: Akerlof's Story as a Graph



Node v_i = reaching day i without sending the package.

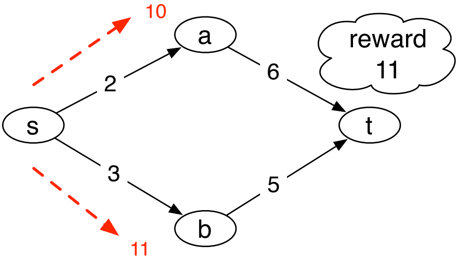
Paths with Rewards



Variation: agent only continues on path if cost $\leq$ reward at t.

- Can model abandonment: agent stops partway through a completed path.
- Can model benefits of choice reduction: deleting nodes can sometimes make graph become traversable.

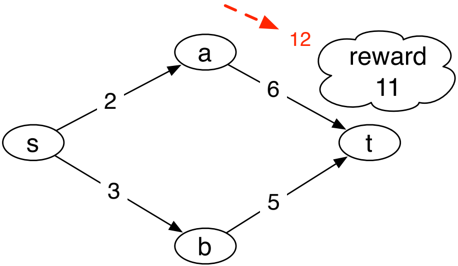
Paths with Rewards



Variation: agent only continues on path if cost $\leq$ reward at t.

- Can model abandonment: agent stops partway through a completed path.
- Can model benefits of choice reduction: deleting nodes can sometimes make graph become traversable.

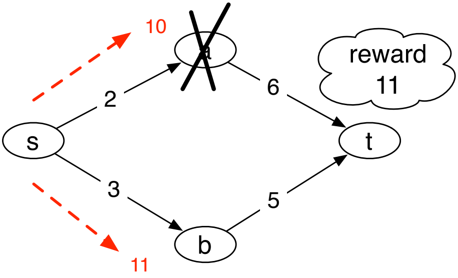
Paths with Rewards



Variation: agent only continues on path if cost $\leq$ reward at t.

- Can model abandonment: agent stops partway through a completed path.
- Can model benefits of choice reduction: deleting nodes can sometimes make graph become traversable.

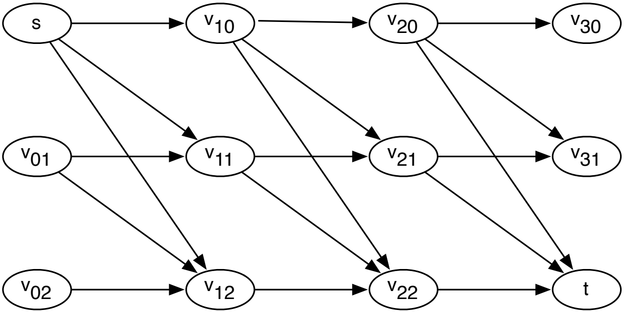
Paths with Rewards



Variation: agent only continues on path if cost $\leq$ reward at t.

- Can model abandonment: agent stops partway through a completed path.
- Can model benefits of choice reduction: deleting nodes can sometimes make graph become traversable.

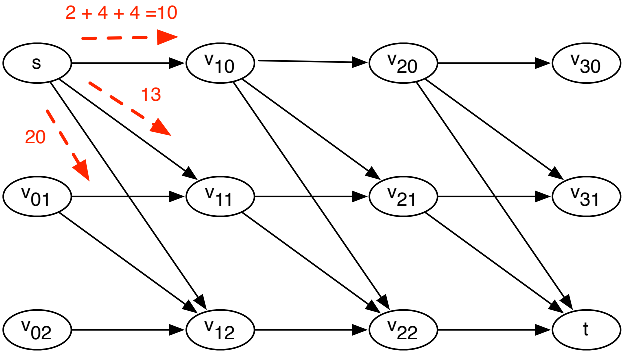
A More Elaborate Example



Three-week short course with two projects.

- Reward of 16 from finishing the course.
- Effort cost in a given week: 1 from doing no project, 4 from doing one, 9 from doing both.
- v_{ij} = the state in which i weeks of the course are done and the student has completed j projects.

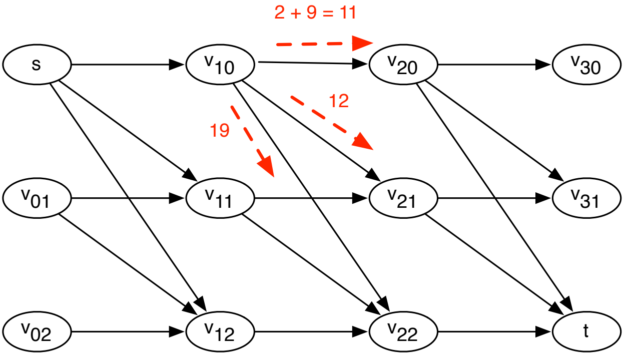
A More Elaborate Example



Three-week short course with two projects.

- Reward of 16 from finishing the course.
- Effort cost in a given week: 1 from doing no project, 4 from doing one, 9 from doing both.
- v_{ij} = the state in which *i* weeks of the course are done and the student has completed *j* projects.

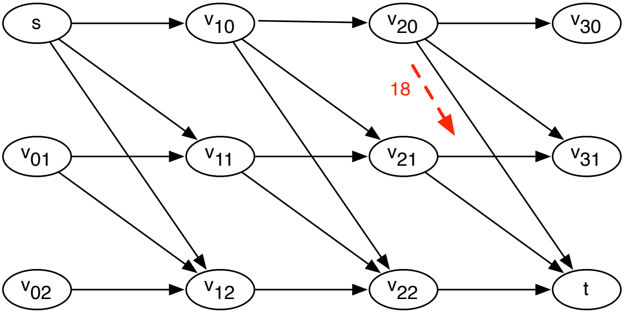
A More Elaborate Example



Three-week short course with two projects.

- Reward of 16 from finishing the course.
- Effort cost in a given week: 1 from doing no project, 4 from doing one, 9 from doing both.
- v_{ij} = the state in which i weeks of the course are done and the student has completed j projects.

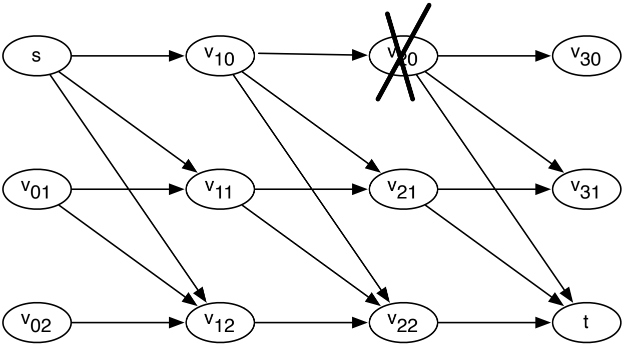
A More Elaborate Example



Three-week short course with two projects.

- Reward of 16 from finishing the course.
- Effort cost in a given week: 1 from doing no project, 4 from doing one, 9 from doing both.
- v_{ij} = the state in which i weeks of the course are done and the student has completed j projects.

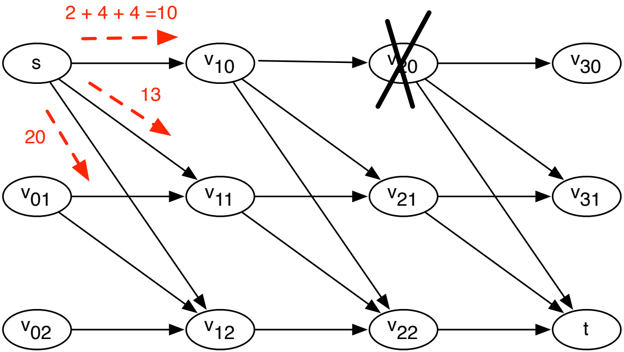
A More Elaborate Example



Three-week short course with two projects.

- Reward of 16 from finishing the course.
- Effort cost in a given week: 1 from doing no project, 4 from doing one, 9 from doing both.
- v_{ij} = the state in which i weeks of the course are done and the student has completed j projects.

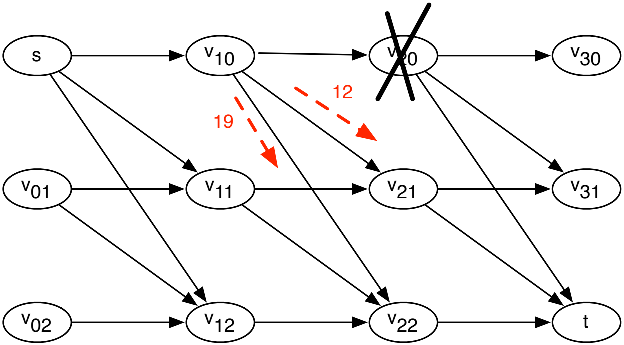
A More Elaborate Example



Three-week short course with two projects.

- Reward of 16 from finishing the course.
- Effort cost in a given week: 1 from doing no project, 4 from doing one, 9 from doing both.
- v_{ij} = the state in which i weeks of the course are done and the student has completed j projects.

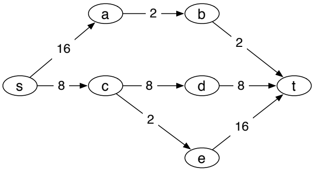
A More Elaborate Example

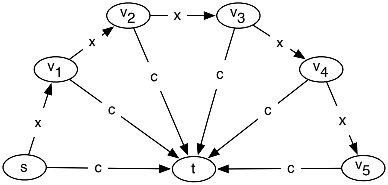


Three-week short course with two projects.

- Reward of 16 from finishing the course.
- Effort cost in a given week: 1 from doing no project, 4 from doing one, 9 from doing both.
- v_{ij} = the state in which i weeks of the course are done and the student has completed j projects.

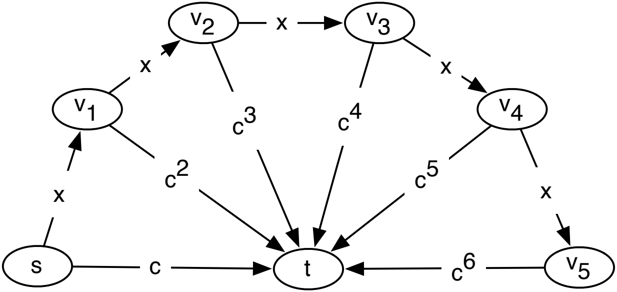
Overview





1. Analyzing present-biased behavior via shortest-path problems.
2. Characterizing instances with high cost ratios.
3. Algorithmic problem: optimal choice reduction to help present-biased agents complete tasks.
4. Heterogeneity: populations with diverse values of b.

A Bad Example for the Cost Ratio

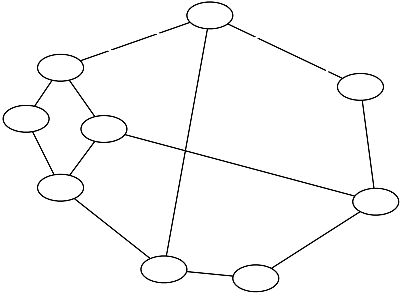


Cost ratio can be roughly b^n, and this is essentially tight.
($n = \#$nodes.)

Can we characterize the instances with exponential cost ratio?

- Goal, informally stated: Must any instance with large cost ratio contain Akerlof's story as a sub-structure?

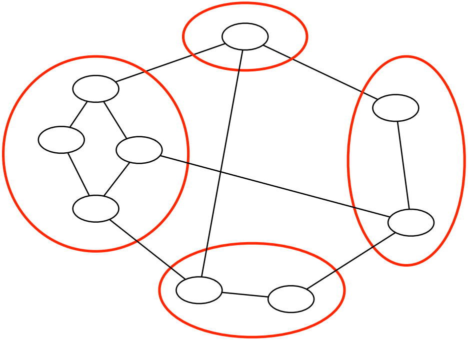
Characterizing Bad Instances via Graph Minors



Graph H is a *minor* of graph G if
we can contract connected subsets of G into "super-nodes"
so as to produce a copy of H.

- In the example: G has a K_4-minor.

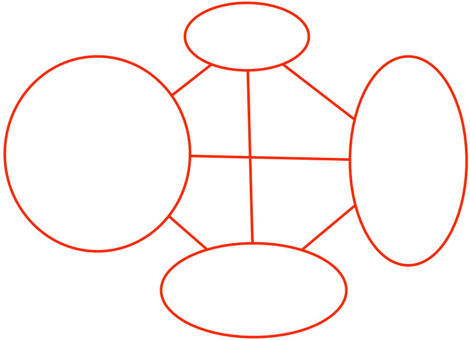
Characterizing Bad Instances via Graph Minors



Graph H is a *minor* of graph G if
we can contract connected subsets of G into "super-nodes"
so as to produce a copy of H.

- In the example: G has a K_4-minor.

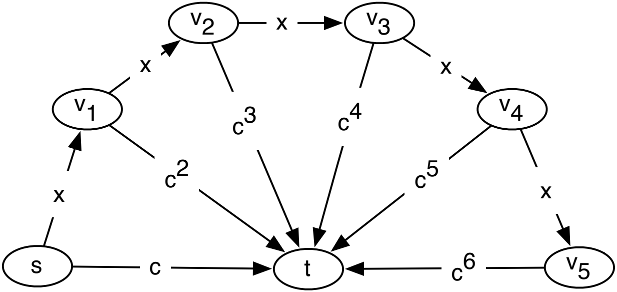
Characterizing Bad Instances via Graph Minors



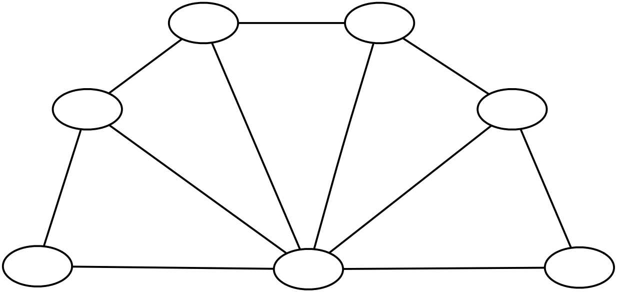
Graph H is a *minor* of graph G if
we can contract connected subsets of G into "super-nodes"
so as to produce a copy of H.

- In the example: G has a K_4-minor.

Characterizing Bad Instances via Graph Minors

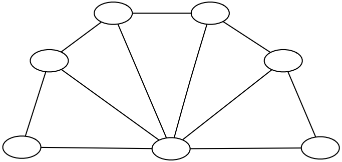


Characterizing Bad Instances via Graph Minors



Characterizing Bad Instances via Graph Minors

The k-fan $\mathcal{F}_k$: the graph consisting of a k-node path, and one more node that all others link to.

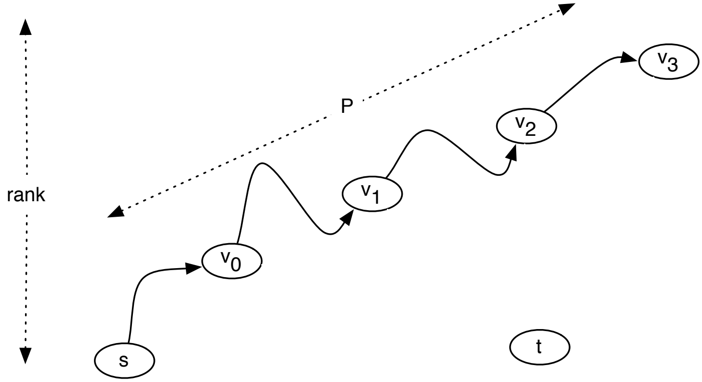


Theorem

For every $\lambda > 1$ there exists $\varepsilon > 0$ such that
if the cost ratio is $> \lambda^n$,
then the underlying undirected graph of the instance contains an $\mathcal{F}_k$-minor for $k = \varepsilon n$.

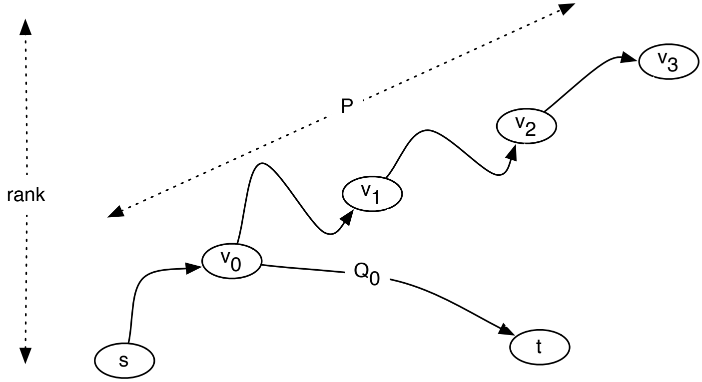
In subsequent work, tight bound by Tang et al 2015.

Sketch of the Proof



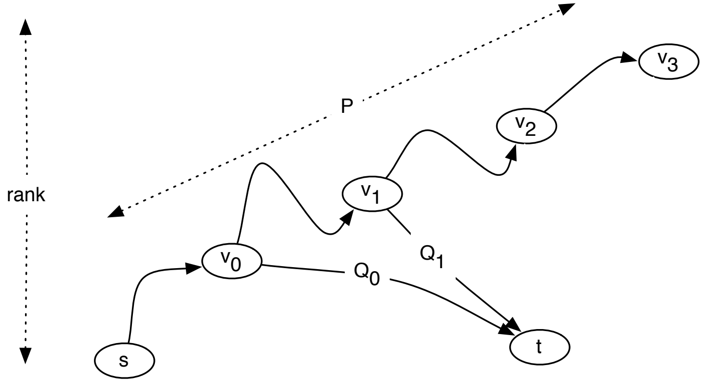
- The agent traverses a path P as it tries to reach t.
- Let the *rank* of a node on P be the logarithm of its dist. to t.
- Show that every time the rank increases by 1, we can construct a new path to t that avoids the traversed path P.

Sketch of the Proof



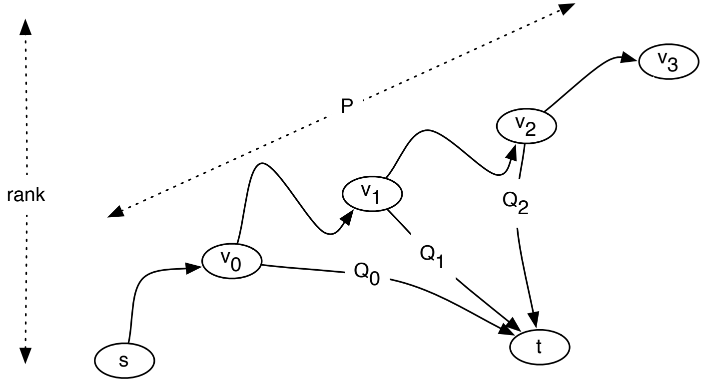
- The agent traverses a path P as it tries to reach t.
- Let the *rank* of a node on P be the logarithm of its dist. to t.
- Show that every time the rank increases by 1, we can construct a new path to t that avoids the traversed path P.

Sketch of the Proof



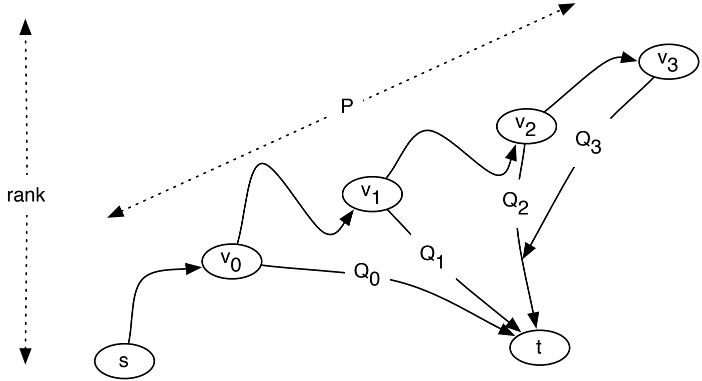
- The agent traverses a path P as it tries to reach t.
- Let the *rank* of a node on P be the logarithm of its dist. to t.
- Show that every time the rank increases by 1, we can construct a new path to t that avoids the traversed path P.

Sketch of the Proof



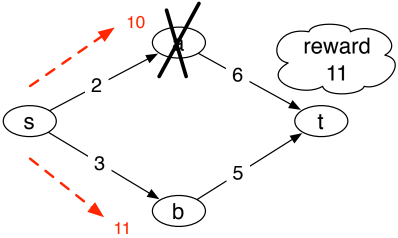
- The agent traverses a path P as it tries to reach t.
- Let the *rank* of a node on P be the logarithm of its dist. to t.
- Show that every time the rank increases by 1, we can construct a new path to t that avoids the traversed path P.

Sketch of the Proof



- The agent traverses a path P as it tries to reach t.
- Let the *rank* of a node on P be the logarithm of its dist. to t.
- Show that every time the rank increases by 1, we can construct a new path to t that avoids the traversed path P.

Choice Reduction



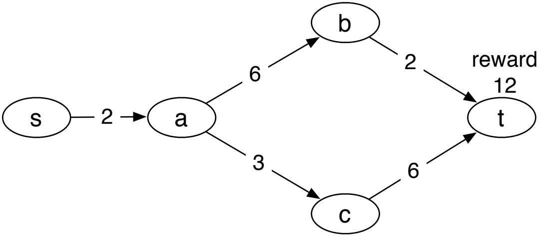
Choice reduction problem: Given G, not traversable by an agent, is there a subgraph of G that is traversable?

- Our initial idea: if there is a traversable subgraph in G, then there is a traversable subgraph that is a path.
- But this is not the case.

Results:

- A characterization of the structure of minimal traversable subgraphs.
- NP-completeness [Feige 2014, Tang et al 2015]
- Open: Approximation by slightly increasing reward and deleting nodes?

Choice Reduction



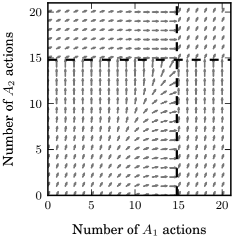
Choice reduction problem: Given G, not traversable by an agent, is there a subgraph of G that is traversable?

- Our initial idea: if there is a traversable subgraph in G, then there is a traversable subgraph that is a path.
- But this is not the case.

Results:

- A characterization of the structure of minimal traversable subgraphs.
- NP-completeness [Feige 2014, Tang et al 2015]
- Open: Approximation by slightly increasing reward and deleting nodes?

Further Questions



Reward systems are a key part of the design space.

- Where does the value reside in rewards for long-range planning? Social, motivational, transactional, ... ?
- Sophisticated agents: aware of their own time-inconsistency
  [O'Donoghue-Rabin 1999]
  How to incorporate sophisticated agents in graph-theoretic model?
  [Kleinberg-Oren-Raghavan, 2015]
- Multi-player settings: interactions between agents with varying levels of bias and sophistication.
- Connect these ideas back to models and data for badge design.
  [Easley-Ghosh13, Anderson et al 13, Immorlica et al 15]