Competition Alleviates Present Bias in Task Completion

Aditya Saraf, Anna Karlin, and Jamie Morgenstern

University of Washington

WINE'20
November 27, 2020

Present Bias: A Shipping Story

You need to ship an item you just sold to the buyer..

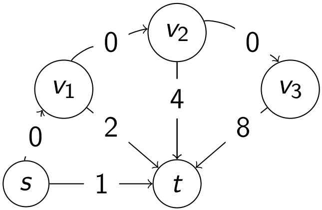
Present Bias: A Shipping Story

You need to ship an item you just sold to the buyer..
but you'll do it tomorrow.

(Naive) Present Bias in Task Completion

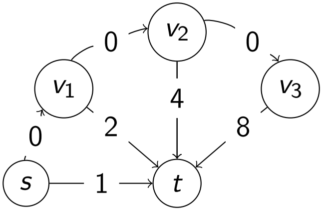
Model tasks as a directed, acyclic graph with start node s and end node t [Kleinberg and Oren, 2014].

Shipping example:



(Naive) Present Bias in Task Completion

- (Biased) agents reason *locally*: they multiply cost of next edge by b
- (Biased) agents reason *naively*: they assume they will behave optimally in the future



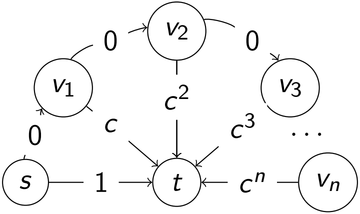
(Naive) Present Bias in Task Completion

n-fan: exponential *cost ratio* ($\frac{\text{biased cost}}{\text{optimal cost}}$)

Theorem

A graph has an exponential cost ratio if and only if it has a (large) fan-like structure.

- $\Longleftarrow$ [Kleinberg and Oren, 2014]
- $\Longrightarrow$ [Tang et al., 2017]



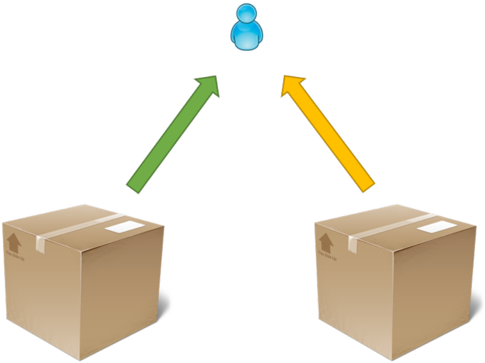
Related Work

Abandonment model:

- task designers can delete parts of the graph $\implies$ NP-hard to find *motivating subgraphs* [Tang et al., 2017]
- task designers can place rewards on intermediate vertices $\implies$ NP-hard to allocate rewards [Albers and Kraft, 2019, Tang et al., 2017]

[Kleinberg et al., 2016] investigates tools that *sophisticated* biased agents can use to limit their harms.

Present Bias: A *Competitive* Shipping Story



With competition:

You need to ship an item you just sold to the buyer...
and *you'll do it today*.

Motivating Competition

Competition:

- happens "naturally" (i.e. without a task designer). Example: two companies (with present bias) competing to get to market first.

Motivating Competition

Competition:

- happens “naturally” (i.e. without a task designer). Example: two companies (with present bias) competing to get to market first.
- is more flexible than deadlines in “designed tasks”. Example: rewarding the first project submission with extra credit.

Competitive Task Planning

- Two agents (A_1, A_2) with identical, public biases b are competing to finish a task first. The winner gets reward r, which is evenly split on a tie.
- For a path, the sum of the weights represent the cost, while the number of edges represent time.

  One edge/hop $=$ one unit of time

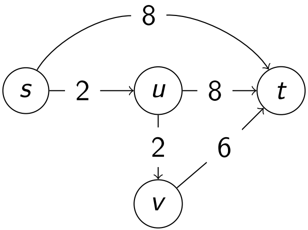
Competitive Task Planning

- Two agents (A_1, A_2) with identical, public biases b are competing to finish a task first. The winner gets reward r, which is evenly split on a tie.
- For a path, the sum of the weights represent the cost, while the number of edges represent time.

  One edge/hop = one unit of time
- Optimal behavior for A_1 at u: depends on A_2's path (length) and A_1's path (length) to u.
- Given A_2's path and A_1's path to u, A_1 should take the best of the cheapest winning, tying, and losing paths:
  $\min(c_w - r, c_t - r/2, c_\ell)$

Competitive Task Planning: Example

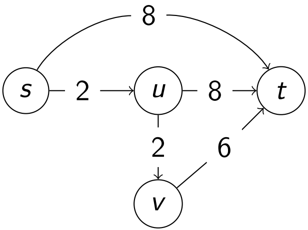
Suppose the agents have bias $b = 2$. Does a reward of $r = 5$ induce a Nash equilibrium on (s, u, t)?



- Assume A_2 takes (s, u, t). At u, the *optimal* path to t is (u, t), since $8 - 2.5 = 5.5 < 8 = 2 + 6$.

Competitive Task Planning: Example

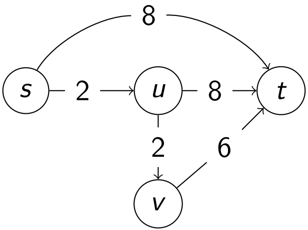
Suppose the agents have bias $b = 2$. Does a reward of $r = 5$ induce a Nash equilibrium on (s, u, t)?



- Assume A_2 takes (s, u, t). At u, the *optimal* path to t is (u, t), since $8 - 2.5 = 5.5 < 8 = 2 + 6$.
- So, the perceived cost of going to u from s is $2 \cdot 2 + 5.5 = 9.5$. The perceived cost of (s, t) at s is $8 \cdot 2 - 5 = 11$.

Competitive Task Planning: Example

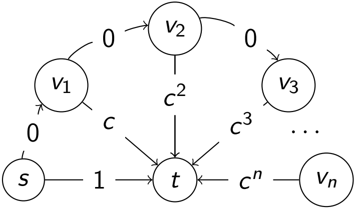
Suppose the agents have bias $b = 2$. Does a reward of $r = 5$ induce a Nash equilibrium on (s, u, t)?



- Assume A_2 takes (s, u, t). At u, the *optimal* path to t is (u, t), since $8 - 2.5 = 5.5 < 8 = 2 + 6$.
- So, the perceived cost of going to u from s is $2 \cdot 2 + 5.5 = 9.5$. The perceived cost of (s, t) at s is $8 \cdot 2 - 5 = 11$.
- At u, the biased agent perceives (u, t) as costing $2 \cdot 8 - 2.5 = 13.5$, while the path through v costs $2 \cdot 2 + 6 = 10$

Overview of Results

1. For graphs with a dominant path[1], a small competitive reward encourages optimal behavior.



2. We provide an algorithm that works on general graphs:
   - inputs: graph G, path Q, bias b
   - output: minimum r that ensures a Nash equilibrium on Q, or $\perp$ if no r works

[1] a cheapest path that is the unique quickest path

Main Theorem

Theorem

Suppose G is a task graph that has a **dominant** *path, O. Then, a reward of $r \geq 2b \cdot \max_{e \in O} c(e)$ guarantees a Nash equilibrium on O, for two agents with bias b. $c(e)$ denotes the cost of edge e.*

Assume A_2 takes O. What is the *optimal* path from an arbitrary node v?

Main Theorem

Theorem

Suppose G is a task graph that has a **dominant** *path, O. Then, a reward of $r \geq 2b \cdot \max_{e \in O} c(e)$ guarantees a Nash equilibrium on O, for two agents with bias b. $c(e)$ denotes the cost of edge e.*

Assume A_2 takes O. What is the *optimal* path from an arbitrary node v?

1. If you haven't deviated from O, and $v \in O$, remain on O. In other words, take the **cheapest path** from v to t.

Main Theorem

Theorem

Suppose G is a task graph that has a **dominant** *path, O. Then, a reward of $r \geq 2b \cdot \max_{e \in O} c(e)$ guarantees a Nash equilibrium on O, for two agents with bias b. $c(e)$ denotes the cost of edge e.*

Assume A_2 takes O. What is the *optimal* path from an arbitrary node v?

1. If you haven't deviated from O, and $v \in O$, remain on O. In other words, take the **cheapest path** from v to t.
2. If you've already deviated from O, or you're going to deviate ($v \notin O$), take the **cheapest path** from v to t.

Proof Sketch

Let $d(v)$ denote the cost of the cheapest path from $v \to t$. Go by induction on the path, at each step determining a sufficient reward to keep A_1 on O.

- They are at $o \in O$, next vertex on $O = o'$. They consider deviating to v.
- They lose if they deviate, and tie if they don't deviate.
- If $bc(o, v) + d(v) \geq bc(o, o') + d(o') - r/2$, they aren't better off deviating.

Proof Sketch

- Need $r/2 \geq b\big(c(o, o') - c(o, v)\big) + d(o') - d(v)$

Proof Sketch

- Need $r/2 \geq b\big(c(o,o') - c(o,v)\big) + d(o') - d(v)$
- Claim: $bc(o,o') \geq bc(o,o') - bc(o,v) + d(o') - d(v)$

  Because: $d(v) + bc(o,v) \geq d(o') + c(o,o')$ by optimality of O

  $\geq d(o')$
- Suffices for $r \geq 2bc(o,o')$
- Quantify over edges $\implies r \geq 2b \max_{e \in O} c(e)$

Problem Definition

- Given: graph G, path $Q \in G$, and bias b
- Output: minimum r that induces a Nash equilibrium on Q (or $\perp$ if no r works)

Naive Approach

Theorem proof suggests naive approach:

1. Start with a reward of 0, step along $u \in Q$, compute the minimum reward to ensure A_1 stays on Q for one more edge.
2. If A_1 wants to deviate to a quicker/tied path, return $\perp$
3. After one pass, do a second pass with the final r

Lemma

Step 1 can be computed efficiently.

False Intuitions

Higher rewards need not encourage quicker paths:

Property (1) (FALSE)

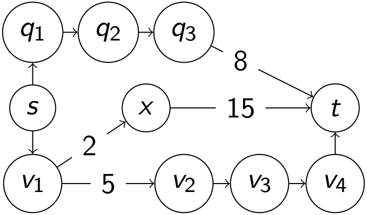
If a reward r guarantees a Nash equilibrium on some path Q, any reward $r' > r$ will either (a), still result in a Nash equilibrium on Q, or (b), cause an agent to deviate to a *quicker* path Q'.

Property (2) (FALSE)

Fix A_2's path. Suppose A_1's best response when the reward is r is some path X. Increasing the reward will not cause their best response to be slower than X.

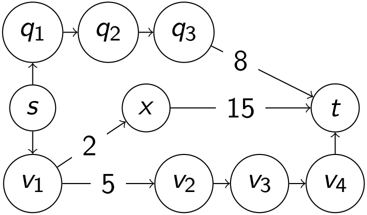
Counterexample to Property (2)

Assume A_2 takes Q, and $b = 10$.



Counterexample to Property (2)

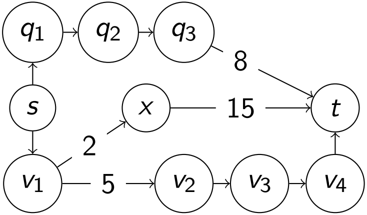
Assume A_2 takes Q, and $b = 10$.



- With $r = 2$, the optimal path from $v_1 \to t$ is to lose on V.
- (Perceived) cost of $Q = 8 - 1$, cost of $V = 5$.

Counterexample to Property (2)

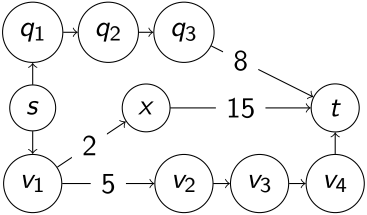
Assume A_2 takes Q, and $b = 10$.



- With $r = 2$, the optimal path from $v_1 \to t$ is to lose on V.
- (Perceived) cost of $Q = 8 - 1$, cost of $V = 5$.
- At v_1, perceived cost of $X = 35 - 2$, perceived cost of $V = 50$.

Counterexample to Property (2)

Assume A_2 takes Q, and $b = 10$.



- With $r = 2$, the optimal path from $v_1 \to t$ is to lose on V.
- (Perceived) cost of $Q = 8 - 1$, cost of $V = 5$.
- At v_1, perceived cost of $X = 35 - 2$, perceived cost of $V = 50$.
- With $r = 10$, opt path from v_1 is to lose on V.
- But now cost of $Q = 8 - 5$, cost of $V = 5$.

Algorithm Outline

- Suppose A_2 takes Q
- $(u, v) \in Q$. $v' \neq v$ is neighbor of u

Algorithm Outline

- Suppose A_2 takes Q
- $(u, v) \in Q$. $v' \neq v$ is neighbor of u
- Assume that optimal path from v to t can be a winning, tying, or losing path as r varies
  - Because of optimality, reward monotonicity holds

Algorithm Outline

- Suppose A_2 takes Q
- $(u, v) \in Q$. $v' \neq v$ is neighbor of u
- Assume that optimal path from v to t can be a winning, tying, or losing path as r varies
  - Because of optimality, reward monotonicity holds
- Thresholds $r_1 < r_2$ for v, $s_1 < s_2$ for v'. Example: $0 < s_1 < r_1 < r_2 < s_2$.
- Inside each interval, cost function for v has slope either $0, -r/2, -r$; same for v'. So at most one subinterval where v is preferred

Algorithm Outline

- Suppose A_2 takes Q
- $(u, v) \in Q$. $v' \neq v$ is neighbor of u
- Assume that optimal path from v to t can be a winning, tying, or losing path as r varies
  - Because of optimality, reward monotonicity holds
- Thresholds $r_1 < r_2$ for v, $s_1 < s_2$ for v'. Example: $0 < s_1 < r_1 < r_2 < s_2$.
- Inside each interval, cost function for v has slope either $0, -r/2, -r$; same for v'. So at most one subinterval where v is preferred
- Collect these in $\mathcal{I}_{v,v'} \implies |\mathcal{I}_{v,v'}| \leq 5$
- Quantify over v', then over v doing repeated pairwise intersections

Conclusion

Summary:

- Competitive can be very helpful vs. present bias
- Provably helpful for some graphs (dominant paths)
- Potentially helpful in general: use the algorithm to find out!

Other results:

- Can prove stronger algorithmic results – all Nash equilibria are efficient to characterize
- Bias uncertainty: analyzed n-fan and gave a simple BNE strategy

References I

S. Albers and D. Kraft. Motivating time-inconsistent agents: A computational approach. **Theory of computing systems**, 63(3): 466–487, 2019.

J. Kleinberg and S. Oren. Time-inconsistent planning: a computational problem in behavioral economics. In **Proceedings of the fifteenth ACM conference on Economics and computation**, pages 547–564, 2014.

J. Kleinberg, S. Oren, and M. Raghavan. Planning problems for sophisticated agents with present bias. In **Proceedings of the 2016 ACM Conference on Economics and Computation**, pages 343–360, 2016.

P. Tang, Y. Teng, Z. Wang, S. Xiao, and Y. Xu. Computational issues in time-inconsistent planning. In **Thirty-First AAAI Conference on Artificial Intelligence**, 2017.