How to Make Decisions (Optimally)

Siddhartha Sen
Microsoft Research NYC
AI for Systems

• **Vision:** Infuse AI to optimize cloud infrastructure decisions, while being:
  • Minimally disruptive (agenda: *Harvesting Randomness*)
  • Synergistic with human solutions (agenda: *HAlbrid algorithms*)
  • Safe and reliable (agenda: *Safeguards*)

• **Impact:** Above criteria differentiate us, ensure wider-spread impact

• **Team:**
  • MSR NYC, MSR India
  • Azure: Azure Compute, Azure Frontdoor
  • Universities: Columbia, NYU, Princeton, Yale, UBC, U. Chicago, Cornell
Vision: Safe optimization without disruption

Evaluate alternatives without disrupting?
Roadmap

• A framework for making systematic decisions: Reinforcement Learning

• A way to reason about decisions in the past: Counterfactual Evaluation

• How to make this work in cloud systems?
  • Successes, fundamental obstacles, workarounds
Decisions in the real world

Which policy maximizes my total reward?
Reinforcement learning (RL)

Which policy maximizes my total reward?
Example: online news articles (MSN)
Example: machine health (Azure cloud)

- Wait time before reboot
- Machine, failure history
- Total downtime
Example: commute options

- bike, subway, car
- weather, traffic
- trip time, cost
Example: online dating

match

user, dating hist

length of relationship
Reinforcement learning reflects real life

• Traditional (supervised) machine learning needs the answer as input

\[ x = \text{image} \quad \Rightarrow \quad y = \text{dog, cat, ...} \]

\[(x, y)^*\]

\( y \) gives you the full answer

train a model: \( x \rightarrow y \)
Reinforcement learning reflects real life

• RL interacts with environment, learns from feedback

\[(x, a, r)^*\]

\(a, r\) only gives a partial answer
train a policy: \(x \rightarrow a\)
How to learn in an RL setting?

• Explore to learn about new actions

• Incorporate reward feedback

• Do this systematically! (Humans are not good at this)
Simple example: online news articles

Humans are bad at this

Policy A (Career)

Policy B (Location)

Clicked

Ignored

This is an A/B test!
Simple example: online news articles

RL: richer policy space, richer representation
Aside: Deep Reinforcement Learning!

- Superhuman ability in Go, Chess
- Lots of engineering/tweaking
  - Learning from self-play not new
- Far from AI apocalypse
  - But (opinion): a glimpse of a more subtle, subconscious overtaking
Testing policies online is inefficient

- Costly (prod deployment)
- Risky (live user traffic)
- Slow (split 100% of traffic)
Testing policies online is inefficient

Instead: randomize directly over actions

Problem: randomizing over policies

Collect data first, then evaluate policies after-the-fact
Test policies offline!

Later evaluate career policy:

- Clicked
  - Engineer
  - Texas
  - Male

- Clicked
  - Engineer
  - Seattle
  - Female

- Ignored
  - Engineer
  - Seattle
  - Male

- Clicked
  - Teacher
  - Texas
  - Female
Counterfactual evaluation (testing policies offline)

• Ask “what if” questions about the past: how would this new policy have performed if I had run it?

• Basic idea: Use (randomized) decisions made by a deployed policy to match/evaluate decisions the new policy would make:

\[ \sum_{\text{matches}} r \]

• Problem: deployed policy’s decisions may be biased
Counterfactual evaluation (testing policies offline)

• Ask “what if” questions about the past: how would this new policy have performed if I had run it?

• Basic idea: Use (randomized) decisions made by a deployed policy to match/evaluate decisions the new policy would make:

\[
\sum_{\text{matches}} \frac{r}{p}
\]

• Test many different policies on the same dataset, offline!

Use probabilities to over/underweight decisions
RL + Counterfactual Evaluation

• Very powerful combination: evaluate a billion policies offline, find the best one
  • Exponential boost over online A/B testing

Can we apply this paradigm to cloud systems?
Example: machine health (Azure Compute)

- Wait time before reboot
- Machine, failure history
- Total downtime
Example: TCP config (Azure Frontdoor)
Example: replica selection (Azure LB)

replica to handle request

req, replica loads latency
What if...

• ... we waited a different amount of time before rebooting?
• ... we used different TCP settings on an edge proxy machine?
• ... we sent a request to a different replica?

Counterfactual evaluation!
Counterfactual evaluation in Systems

• Opportunity: Many systems are **naturally randomized**
  • Load balancing, data replicas, cache eviction, fault handling, etc.
  • When we need to spread things, when choices are ambiguous
    ⇒ *Free exploration!*

• Opportunity: Many systems provide **implicit feedback**
  • Naïve defaults, conservative parameter settings
  • Worse settings yield more information
    ⇒ *Free feedback!*
# Counterfactual evaluation in Systems

<table>
<thead>
<tr>
<th>Challenge</th>
<th>Technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mess of methods/techniques spanning multiple disciplines</td>
<td><em>Taxonomy</em></td>
</tr>
<tr>
<td>Huge action spaces (coverage)</td>
<td><em>Spatial coarsening</em></td>
</tr>
<tr>
<td>Stateful, non-independent decisions</td>
<td><em>Temporal coarsening, Time horizons</em></td>
</tr>
<tr>
<td>Dynamic environments</td>
<td><em>(Baseline normalization)</em></td>
</tr>
</tbody>
</table>
Taxonomy for counterfactual evaluation

- Supervised Learning
  - Direct method
- Reinforcement Learning (contextual bandits)
  - Unbiased estimator (DR)
- Reinforcement Learning (general)
  - Unbiased estimator + time horizon (DR-Time)

- Feedback?
  - Yes → Independent decisions?
    - Yes → Unbiased estimator (DR)
    - No → Reinforcement Learning (general)
  - Partial → Randomize/explore
    - Yes → Unbiased estimator + time horizon (DR-Time)
    - No → Reinforcement Learning (general)
- Randomization?
  - Yes
  - No → Randomize/explore
Example: Machine health in Azure Compute

- Wait for some time, then reboot
Example: Machine health in Azure Compute

- Wait for some time, then reboot
- Wait for \{1,2,\ldots,10 \text{ min}\}

Spatial coarsening
Example: Machine health in Azure Compute

<table>
<thead>
<tr>
<th>Decision?</th>
<th>Action</th>
<th>[-]Reward</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine A</td>
<td>Wait 10 min</td>
<td>5 min</td>
</tr>
<tr>
<td>Machine B</td>
<td>Wait 10 min</td>
<td>3 min</td>
</tr>
<tr>
<td>Machine C</td>
<td>Wait 10 min</td>
<td>10 min + reboot</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

...
Example: Machine health in Azure Compute

<table>
<thead>
<tr>
<th>Decision?</th>
<th>Action</th>
<th>[-]Reward</th>
<th>Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine A</td>
<td>Wait 10 min</td>
<td>5 min</td>
<td>Wait 1,2,...,9</td>
</tr>
<tr>
<td>Machine B</td>
<td>Wait 10 min</td>
<td>3 min</td>
<td>Wait 1,2,...,9</td>
</tr>
<tr>
<td>Machine C</td>
<td>Wait 10 min</td>
<td>10 min + reboot</td>
<td>Wait 1,2,...,9</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Example: Machine health in Azure Compute

<table>
<thead>
<tr>
<th>Decision?</th>
<th>Action</th>
<th>[−]Reward</th>
<th>Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine A</td>
<td>Wait 6 min</td>
<td>5 min</td>
<td>Wait 1,2,...,9</td>
</tr>
<tr>
<td>Machine B</td>
<td>Wait 2 min</td>
<td>3 min</td>
<td>Wait 1,2,...,9</td>
</tr>
<tr>
<td>Machine C</td>
<td>Wait 10 min</td>
<td>10 min + reboot</td>
<td>Wait 1,2,...,9</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

...
Example: Machine health in Azure Compute

<table>
<thead>
<tr>
<th>Decision?</th>
<th>Action</th>
<th>[-]Reward</th>
<th>Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine A</td>
<td>Wait 6 min</td>
<td>5 min</td>
<td>Wait 1,2,...,9</td>
</tr>
<tr>
<td>Machine B</td>
<td>Wait 2 min</td>
<td>2 min + reboot</td>
<td>Wait 1</td>
</tr>
<tr>
<td>Machine C</td>
<td>Wait 10 min</td>
<td>10 min + reboot</td>
<td>Wait 1,2,...,9</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Implicit feedback
Results: Machine health in Azure Compute
Results: Machine health in Azure Compute
Example: TCP config in Azure Frontdoor

- TCP parameters:
  - initial cwnd
  - initial RTO
  - min RTO
  - max SYN retransmit
  - delayed ACK freq
  - delayed ACK timeout
Example: TCP config in Azure Frontdoor

- TCP parameters:
  - initial cwnd
  - initial RTO
  - min RTO
  - max SYN retransmit
  - delayed ACK freq
  - delayed ACK timeout

- Pick from 17 different configurations, per hour per machine
Example: TCP config in Azure Frontdoor

- Dynamic workload and environment
- Assign “control” machine to each RL machine as baseline, report delta
Example: TCP config in Azure Frontdoor

- Dynamic workload and environment
- Assign “control” machine to each RL machine as baseline, report delta

Baseline normalization
Results: TCP config in Azure Frontdoor

<table>
<thead>
<tr>
<th>Estimate</th>
<th>Reward</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground truth</td>
<td>0.713</td>
<td>--</td>
</tr>
<tr>
<td>DR</td>
<td>0.720 (0.637, 0.796)</td>
<td>0.97%</td>
</tr>
</tbody>
</table>
Lesson: Unbiased estimator vs. biased policy

<table>
<thead>
<tr>
<th>Configuration</th>
<th>ATA</th>
<th>BOM</th>
<th>FRA</th>
<th>GRU</th>
<th>JNB</th>
<th>MIA</th>
<th>SLA</th>
<th>YTO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 (default)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Lesson: Unbiased estimator vs. biased policy

<table>
<thead>
<tr>
<th>Configuration</th>
<th>ATA</th>
<th>BOM</th>
<th>FRA</th>
<th>GRU</th>
<th>JNB</th>
<th>MIA</th>
<th>SLA</th>
<th>YTO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.20%</td>
<td>3.61%</td>
<td>2.82%</td>
<td>5.34%</td>
<td>0.61%</td>
<td>2.52%</td>
<td>2.34%</td>
<td>4.65%</td>
</tr>
<tr>
<td>2 (default)</td>
<td>3.93%</td>
<td>2.89%</td>
<td>3.76%</td>
<td>2.48%</td>
<td>0.00%</td>
<td>2.34%</td>
<td>4.16%</td>
<td>4.65%</td>
</tr>
<tr>
<td>3</td>
<td>1.75%</td>
<td>3.13%</td>
<td>2.71%</td>
<td>1.72%</td>
<td>1.21%</td>
<td>3.04%</td>
<td>3.12%</td>
<td>6.98%</td>
</tr>
<tr>
<td>4</td>
<td>5.82%</td>
<td>2.89%</td>
<td>3.06%</td>
<td>4.96%</td>
<td>6.06%</td>
<td>5.73%</td>
<td>4.68%</td>
<td>9.30%</td>
</tr>
<tr>
<td>5</td>
<td>1.89%</td>
<td>3.61%</td>
<td>2.94%</td>
<td>3.82%</td>
<td>4.24%</td>
<td>3.39%</td>
<td>2.34%</td>
<td>0.00%</td>
</tr>
<tr>
<td>6</td>
<td>2.04%</td>
<td>4.58%</td>
<td>0.59%</td>
<td>3.05%</td>
<td>1.21%</td>
<td>4.60%</td>
<td>2.34%</td>
<td>2.33%</td>
</tr>
<tr>
<td>7</td>
<td>2.77%</td>
<td>3.86%</td>
<td>5.29%</td>
<td>4.77%</td>
<td>3.03%</td>
<td>3.21%</td>
<td>5.45%</td>
<td>4.65%</td>
</tr>
<tr>
<td>8</td>
<td>15.57%</td>
<td>20.96%</td>
<td>25.76%</td>
<td>21.76%</td>
<td>8.48%</td>
<td>21.88%</td>
<td>18.70%</td>
<td>46.51%</td>
</tr>
<tr>
<td>9</td>
<td>5.68%</td>
<td>3.61%</td>
<td>3.88%</td>
<td>4.20%</td>
<td>0.61%</td>
<td>4.60%</td>
<td>3.38%</td>
<td>2.33%</td>
</tr>
<tr>
<td>10</td>
<td>3.49%</td>
<td>3.37%</td>
<td>4.59%</td>
<td>1.72%</td>
<td>9.70%</td>
<td>3.21%</td>
<td>3.12%</td>
<td>2.33%</td>
</tr>
<tr>
<td>11</td>
<td>4.08%</td>
<td>2.89%</td>
<td>3.76%</td>
<td>1.53%</td>
<td>0.61%</td>
<td>3.91%</td>
<td>3.38%</td>
<td>2.33%</td>
</tr>
<tr>
<td>12</td>
<td>6.11%</td>
<td>3.13%</td>
<td>3.88%</td>
<td>4.39%</td>
<td>2.42%</td>
<td>3.13%</td>
<td>2.60%</td>
<td>2.33%</td>
</tr>
<tr>
<td>13</td>
<td>26.93%</td>
<td>24.34%</td>
<td>22.82%</td>
<td>27.29%</td>
<td>33.94%</td>
<td>25.35%</td>
<td>30.13%</td>
<td>2.33%</td>
</tr>
<tr>
<td>14</td>
<td>4.22%</td>
<td>2.89%</td>
<td>3.53%</td>
<td>3.24%</td>
<td>9.09%</td>
<td>3.99%</td>
<td>4.42%</td>
<td>0.00%</td>
</tr>
<tr>
<td>15</td>
<td>3.64%</td>
<td>7.71%</td>
<td>5.76%</td>
<td>3.82%</td>
<td>5.45%</td>
<td>1.91%</td>
<td>1.82%</td>
<td>2.33%</td>
</tr>
<tr>
<td>16</td>
<td>5.97%</td>
<td>2.89%</td>
<td>1.18%</td>
<td>2.29%</td>
<td>12.12%</td>
<td>2.60%</td>
<td>3.64%</td>
<td>0.00%</td>
</tr>
<tr>
<td>17</td>
<td>2.91%</td>
<td>3.61%</td>
<td>3.65%</td>
<td>3.63%</td>
<td>1.21%</td>
<td>4.60%</td>
<td>4.42%</td>
<td>6.88%</td>
</tr>
</tbody>
</table>
Lesson: Unbiased estimator vs. biased policy

Production policy and DR initially agree

Production policy drifts away

In the end, the Production policy converges to DR

Estimated performance

0 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08

Time (datapoints)

0 100 200 300 400 500 600 700

Production policy
DR
Example: Replica selection

- Choose replica to process each request
Example: Replica selection

- New policy: always send to server 1?
- Problem: non-independent decisions interact over time!
Results: Replica selection

- Run Zipf workload, collect data
- Evaluate new policy: hash(key) % num_replicas
Takeaways

• RL + Counterfactual evaluation is a powerful paradigm; use it to reason about systems!

• Opportunities: natural randomness, implicit feedback

• Challenges: huge action spaces (coverage), non-independent decisions, dynamic environments
  • Fundamental problem: experimenting at small scale/fraction of traffic may not reflect performance at full scale
Challenges in real-life decisions

- Commute options
  - Only one datapoint per day (coverage)
  - Changing traffic patterns, construction, station closures (dynamic environment)
Challenges in real-life decisions

• Online dating
  • Few datapoints, exploration costly \((\text{coverage})\)
  • Very unromantic
Can we (should we) optimize real life?

• RL + counterfactual evaluation is not enough

• Combine with behavioral psychology
  • Model behavior, learn from others, avoid behavioral traps
  • Automate Dan

• But: allow individuality, leverage natural exploration

Dan Goldstein