DATA 61

AB Testing for Process Versions with Contextual Multi-armed Bandit Algorithms

Suhrid Satyal (presenter),

Ingo Weber, Hye-young Paik, Claudio Di Ciccio, Jan Mendling

June 2018

www.csiro.au

UNSW SYDNEY

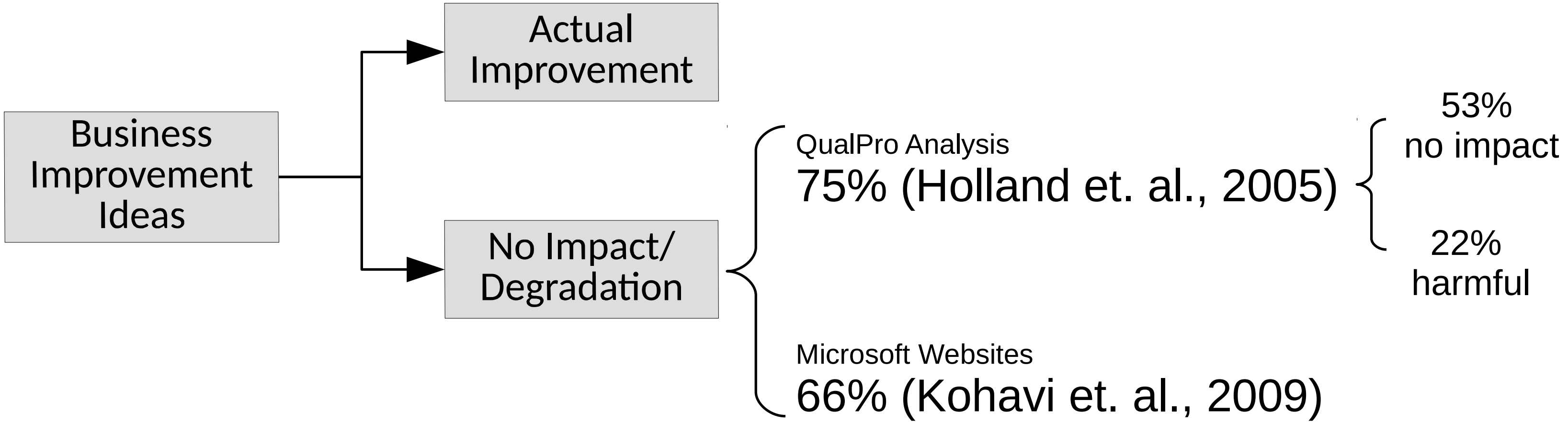
WU WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

CSIRO

Outline

- Motivation
- AB Testing, AB-BPM approach, and their limitations
- Requirements
- Proposed Solution
  - System Architecture
  - ProcessBandit Algorithm
  - Reward Design
- Evaluation
- Conclusion

Motivation



Business Improvement Ideas

Actual Improvement

No Impact/ Degradation

QualPro Analysis
75% (Holland et. al., 2005)

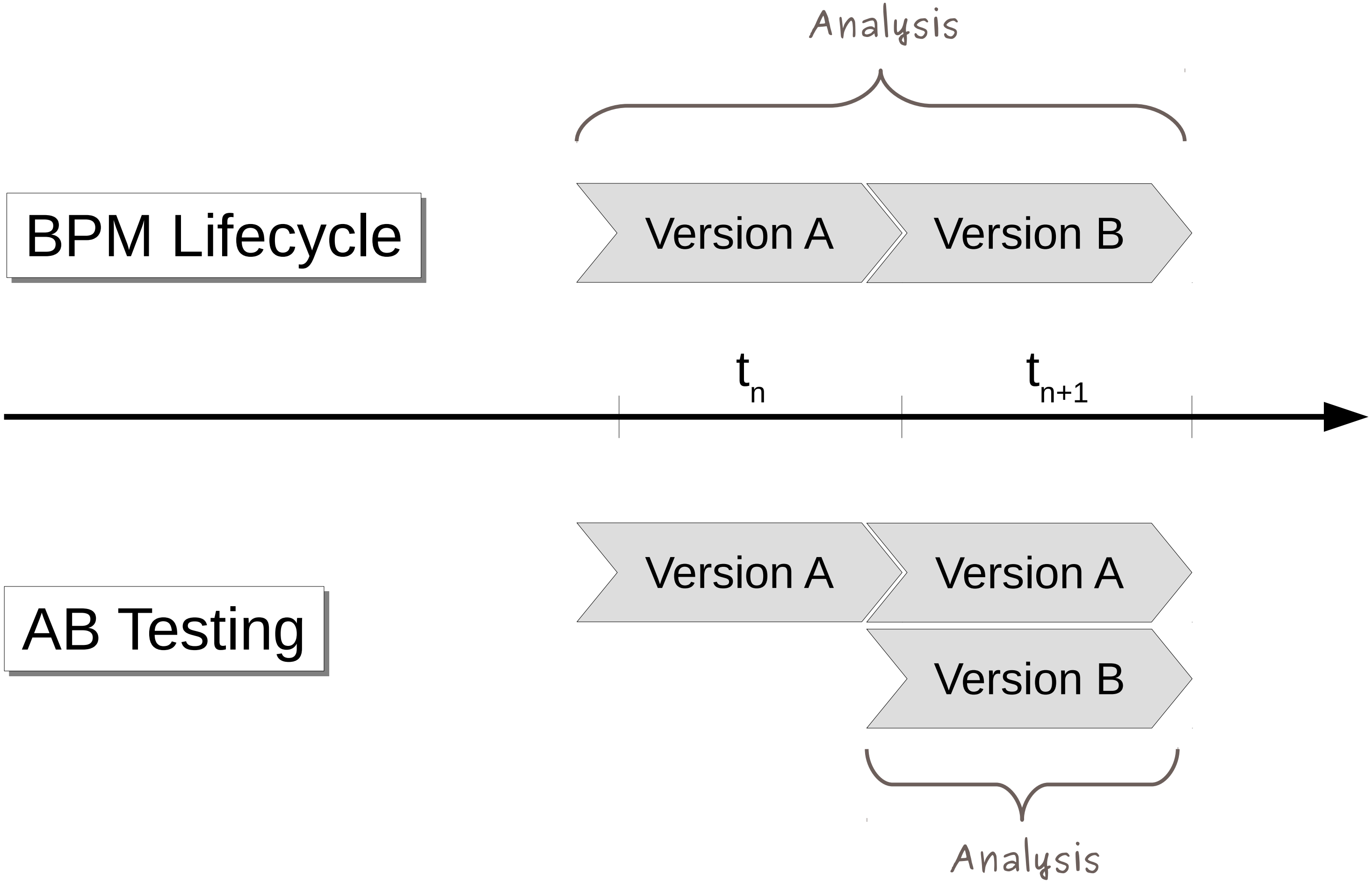
53% no impact

22% harmful

Microsoft Websites
66% (Kohavi et. al., 2009)

- Business Process
  - Chain of Events, Activities, and Decisions
  - Expressed as Process Models
  - Instantiated and Executed by a Process Engine

Improvement Approach



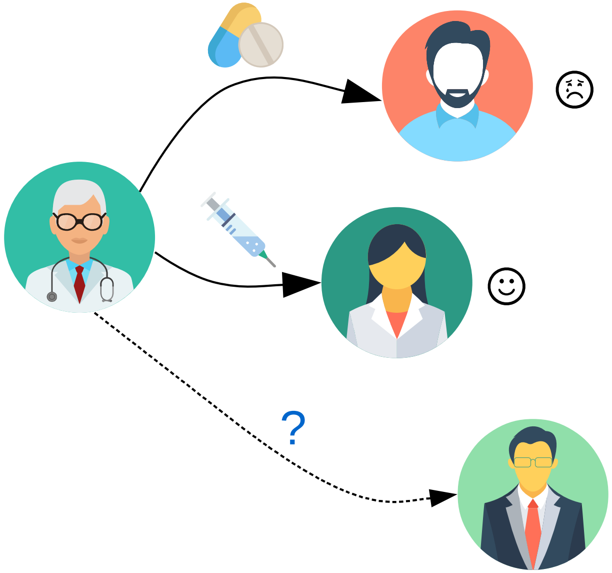
AB Testing

- Randomized Experiment in production – A vs B vs …
- Used to test micro-changes in web applications
- Test fairly, fail fast
- What if the test is risky?

Example of risk management

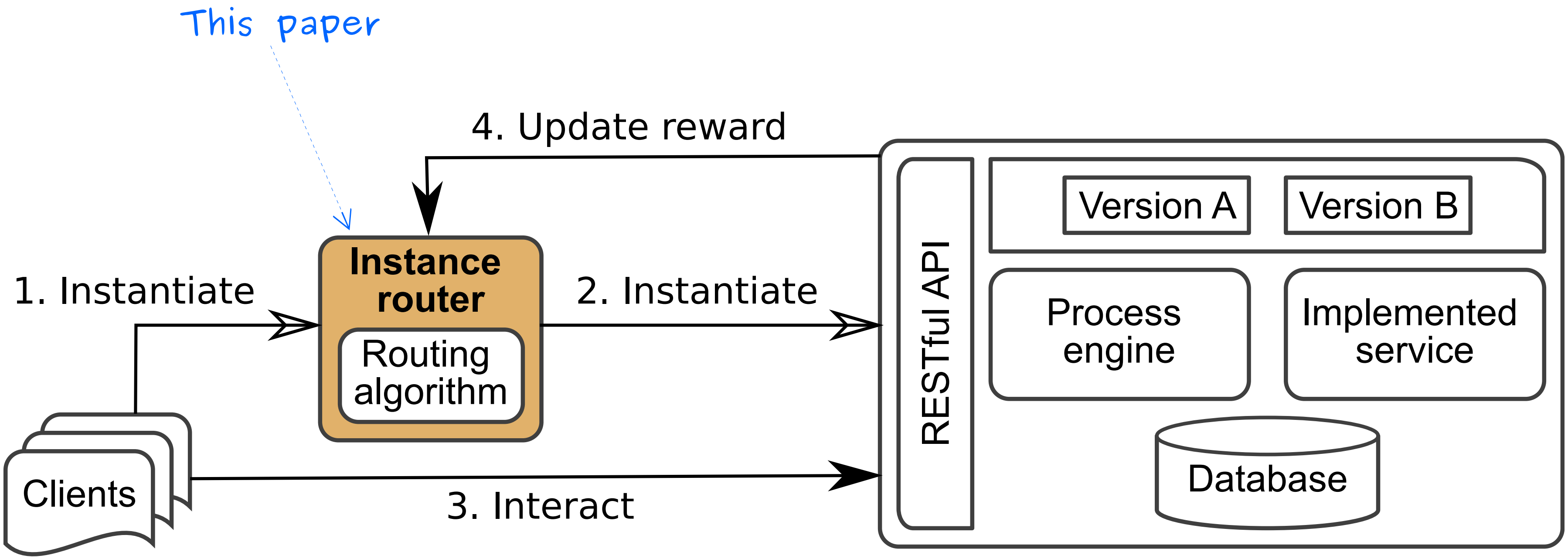
Thompson Sampling in Clinical trials

Routing algorithm



Icons made by Freepik from www.flaticon.com

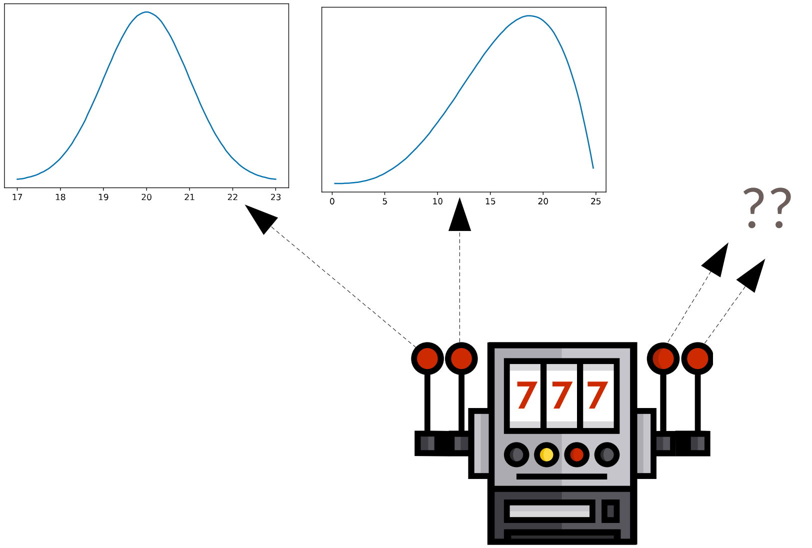
AB-BPM Architecture



Source: Satyal et. al. (2017)

Routing with multi-armed bandits

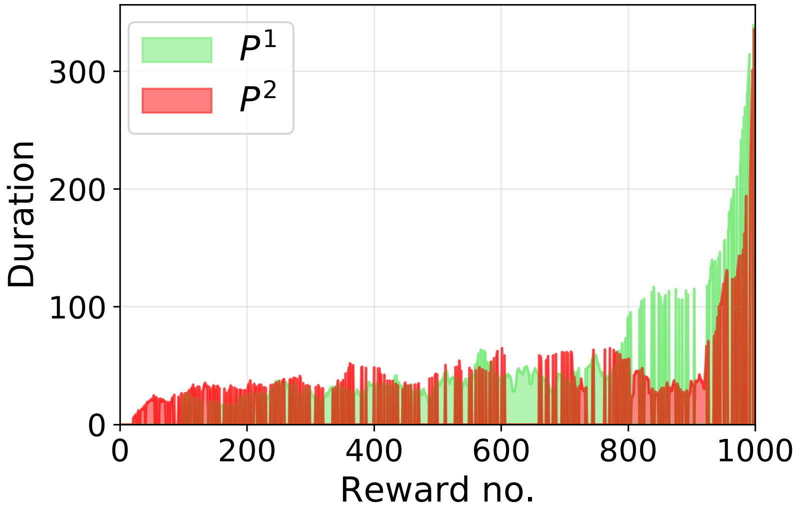
- Contextual multi-armed bandit
- Based on LinUCB (Li et. al. 2010)
- Arm = Process Version
- Phase 1 (P1) – Experimentation (or exploration)
  - Instantiate some process versions at random (decay fxn + LinUCB approach)
  - Observe rewards only for processes instantiated during this phase
- Phase 2 (P2) - Exploitation
  - Do not observe rewards



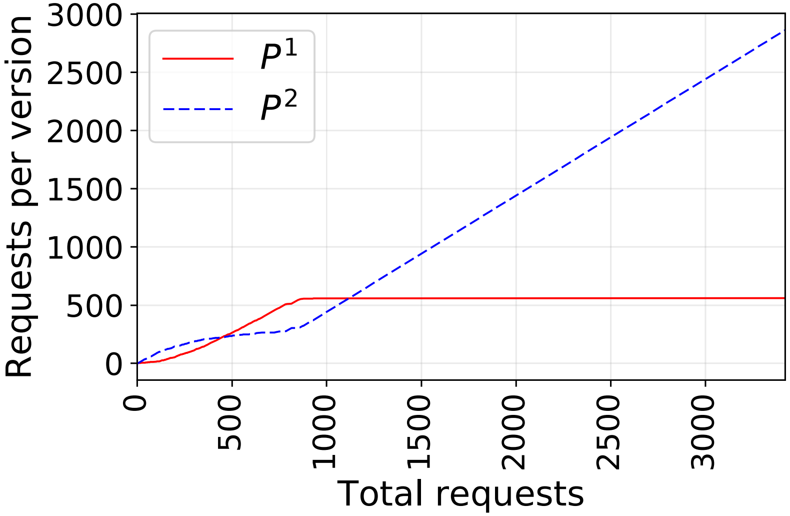
LTAvgR Algorithm and Rewards

- Designed for long running business processes
- One performance indicator (e.g. duration)

Good Rewards come early,
bad ones come late



Send more requests to
Better version over time

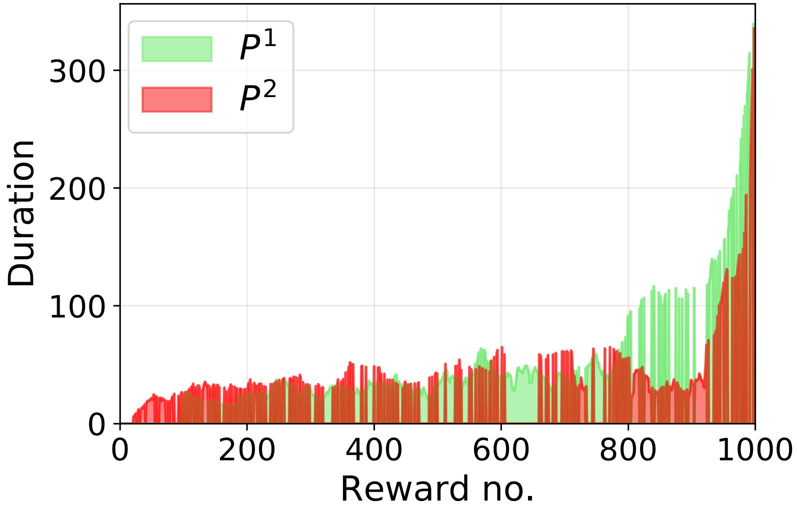


Source: Satyal et. al. (2017)

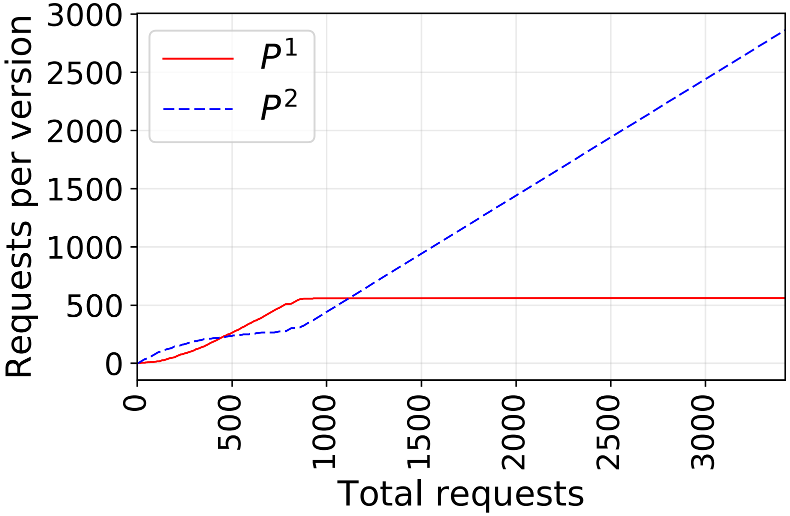
LTAvgR Algorithm and Rewards

- Designed for long running business processes
- One performance indicator (e.g. duration)

Good Rewards come early,
bad ones come late



Send more requests to
Better version over time



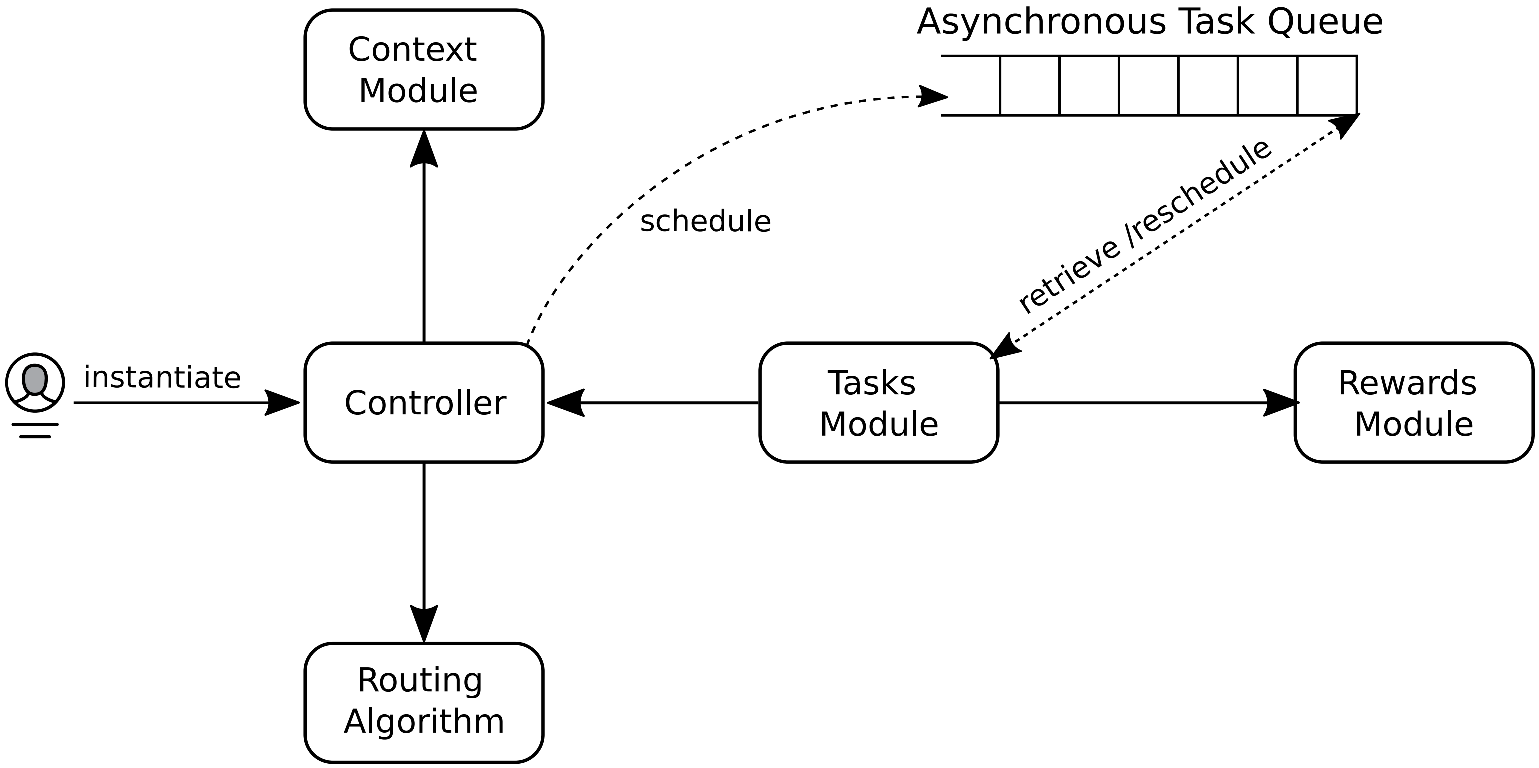
What about complex scenarios?

Source: Satyal et. al. (2017)

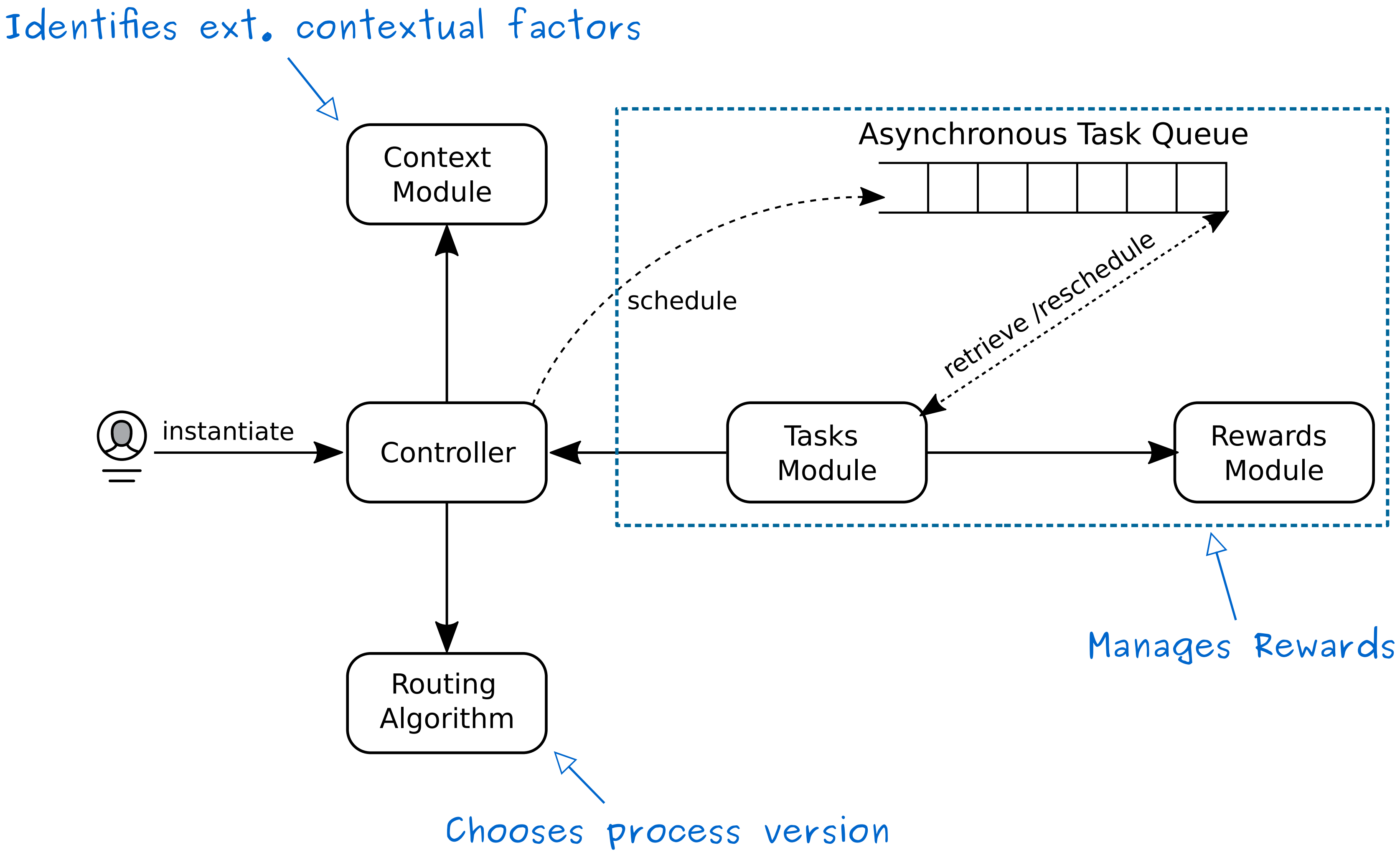
General BPM Scenarios

- Performance is determined by multiple Process Performance Indicators (PPIs) e.g. profit margin & user satisfaction
- Individual PPIs are available at different times
- Most process instances do not provide all PPIs e.g. user satisfaction
- Performance is affected by factors external to the process e.g. Weather Conditions (van der Aalst, et. al. 2007)

Instance Router Architecture

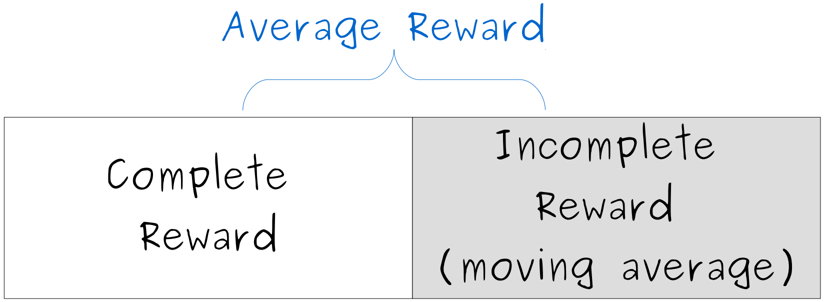


Instance Router Architecture



ProcessBandit Algorithm

- Fetch reward asynchronously
- Apply partial rewards
- Limit no. of incomplete rewards

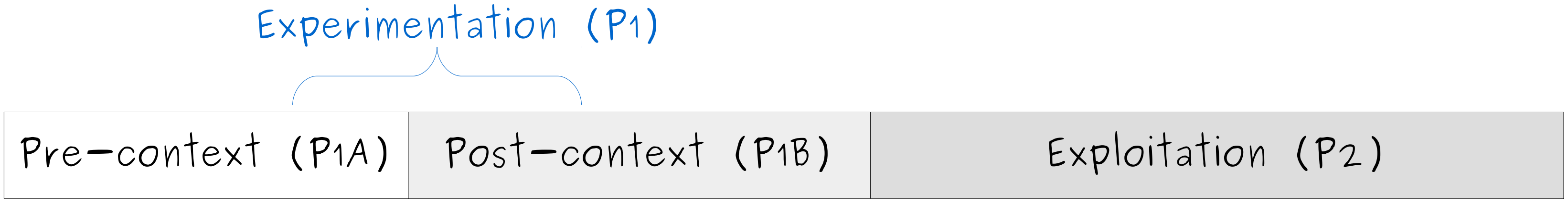


- 3 Phased approach

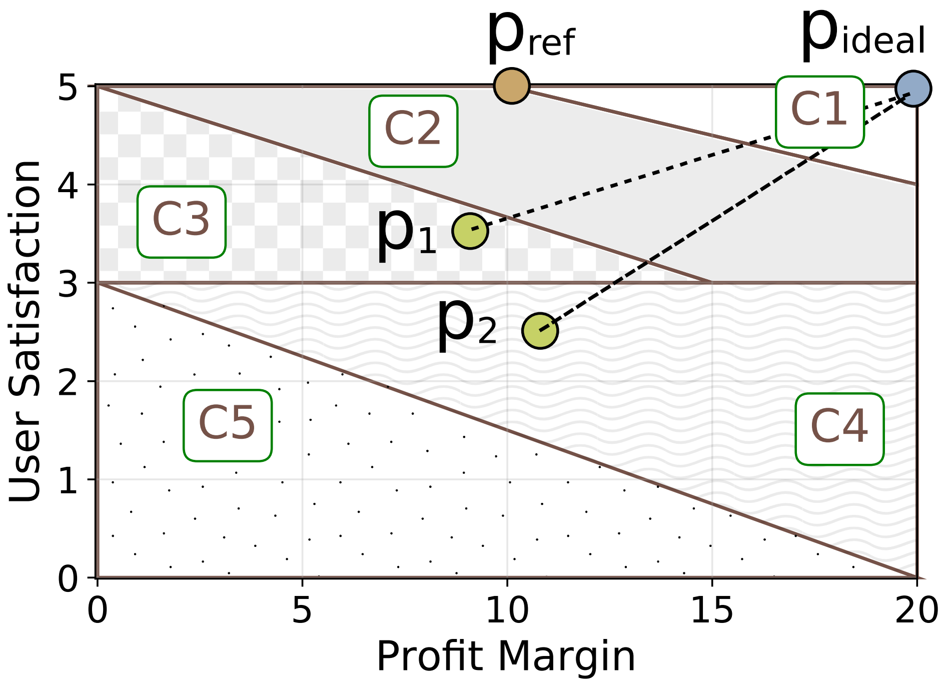
P1A Collect external contextual data, use only request information

P1B Reset algorithm & use contextual data (if necessary)

P2 Send requests to version with best performance



Reward Design



P_i = (pr. margin, user sat.)

P_{ideal} = (best pr. margin, best user sat.)

P_{ref} = (assumed pr. margin, assumed user sat.)

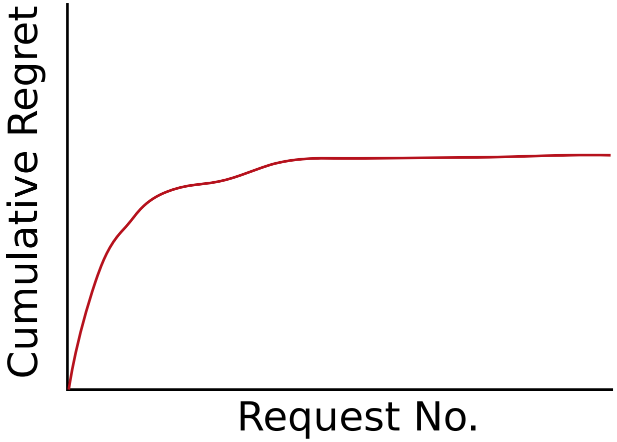
Distance based – implicit, too fine grained
What does dist. P1 vs dist. P2 indicate? ✗

Classification based – Explicit, coarse grained
e.g. C1 is x times better than C2 ✓

Experiment Setup

- Regret
  - reward from best version – observed reward
  - Zero-regret strategy

    avg. regret per request tends to 0



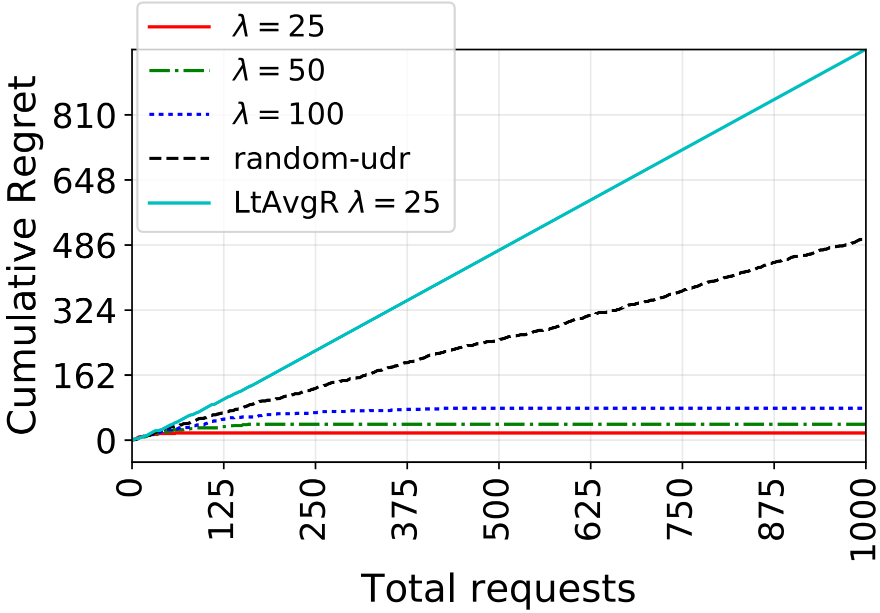
- Best version can be found only if all PPIs, request context, and external contextual factors are available

	Contextual Factor f=1				Contextual Factor f=2			
	Profit Margin		User Satisfaction		Profit Margin		User Satisfaction	
Req. Context	Version A	Version B	Version A	Version B	Version A	Version B	Version A	Version B
X	9	11	3	2.5	11	9	2.5	3
Y	11	9	2.5	3	9	11	3	2.5

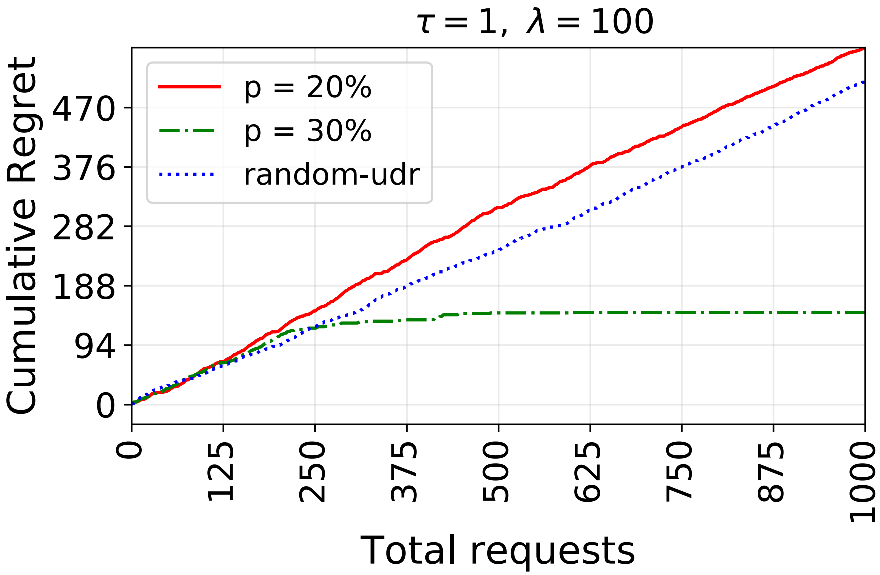
- Distributed Application: Python + Nginx + Redis + Docker

Convergence Characteristics

All PPIs are available



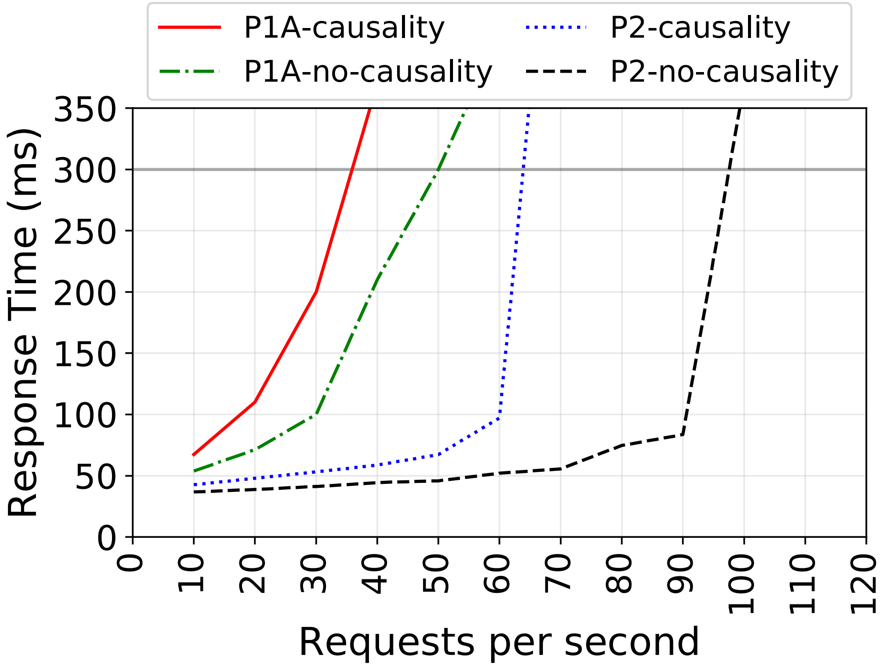
All PPIs are available
Only for p% instances



Scalability

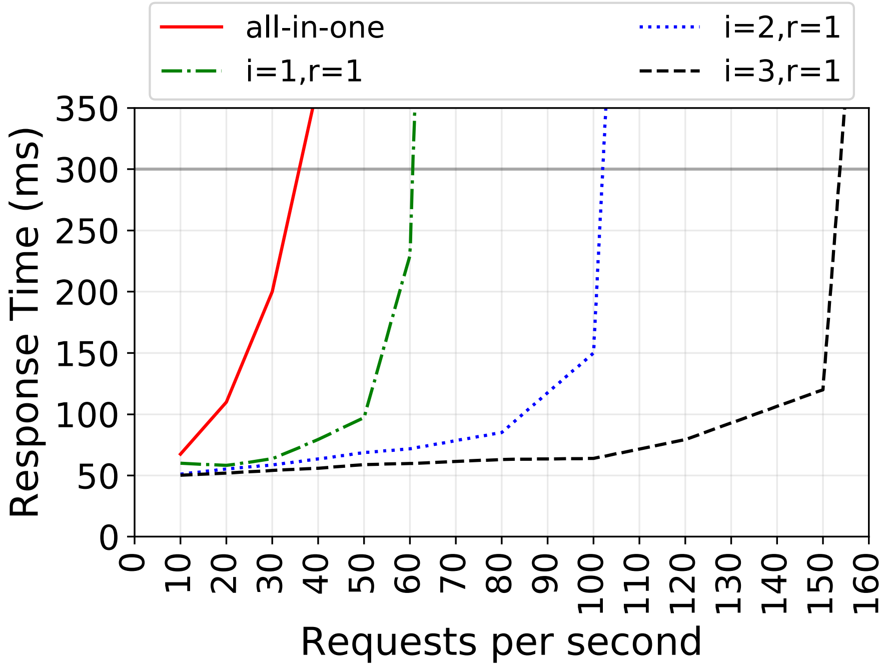
Throughput

Phase-Configuration



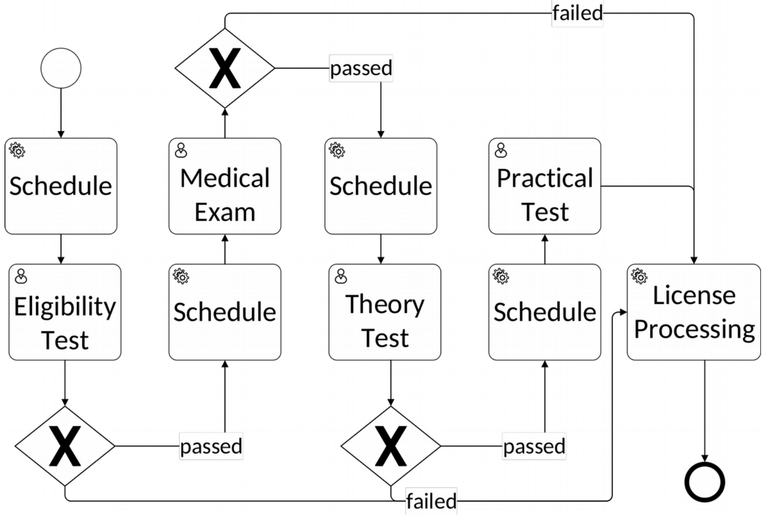
Horizontal Scalability

i = alg. Server, r = redis server

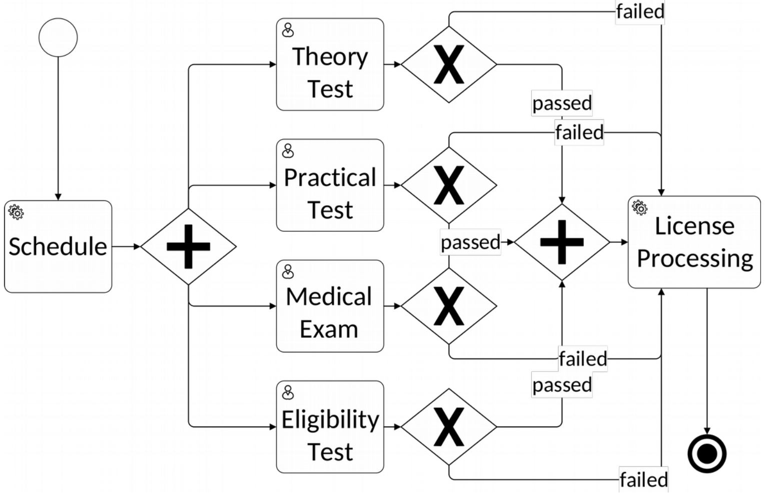


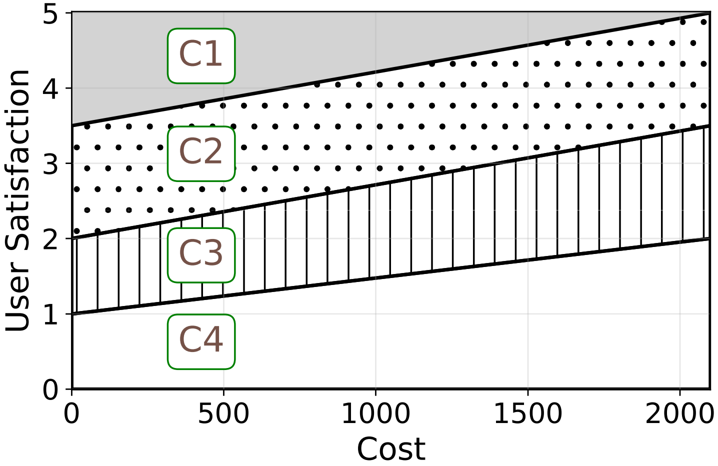
Pilot Licensing Scenario

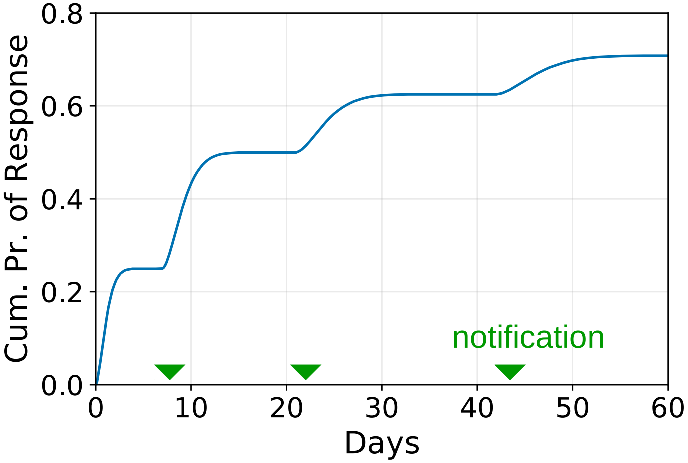
Version A



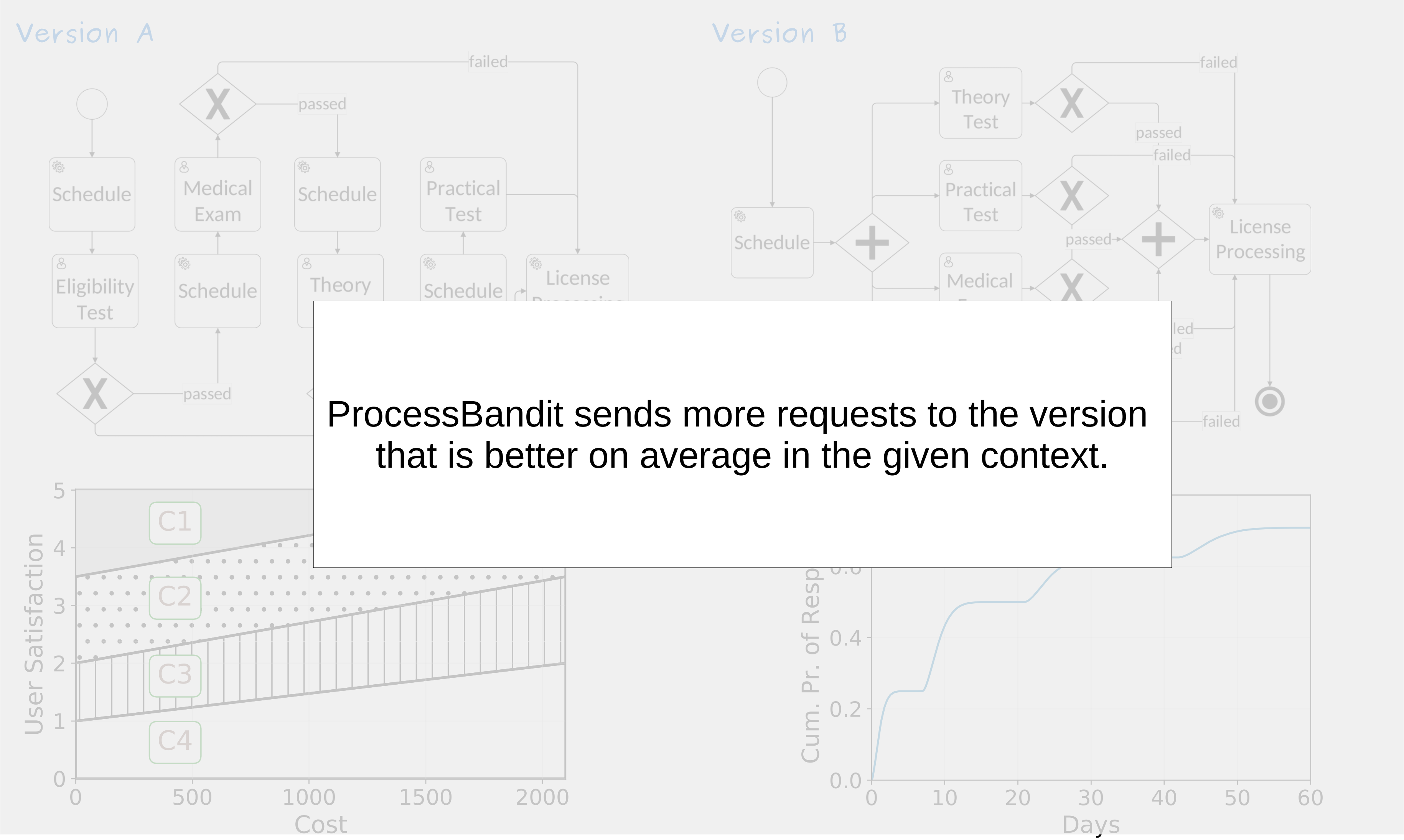
Version B







Pilot Licensing Scenario



ProcessBandit sends more requests to the version that is better on average in the given context.

Conclusion

- AB Testing system and algorithm for generalized BPM scenarios
  - Modular architecture
  - ProcessBandit algorithm
- Part of AB-BPM approach for business process improvement
- Extension of our previous work (BPM 2017)
- Future work – Real world studies, Extension of AB-BPM methodology

Researchgate Project

Thank you

Suhrid Satyal
PhD Candidate
Architecture and Analytics Platforms Team (AAP), Data61
Computer Science and Engineering, UNSW

e Suhrid.Satyal@data61.csiro.au
Australian Technology Park, Lvl 5, 13 Garden Street,
Eveleigh, NSW 2015 l Locked Bag 9013, Alexandria NSW 2015
www.data61.csiro.au

www.csiro.au