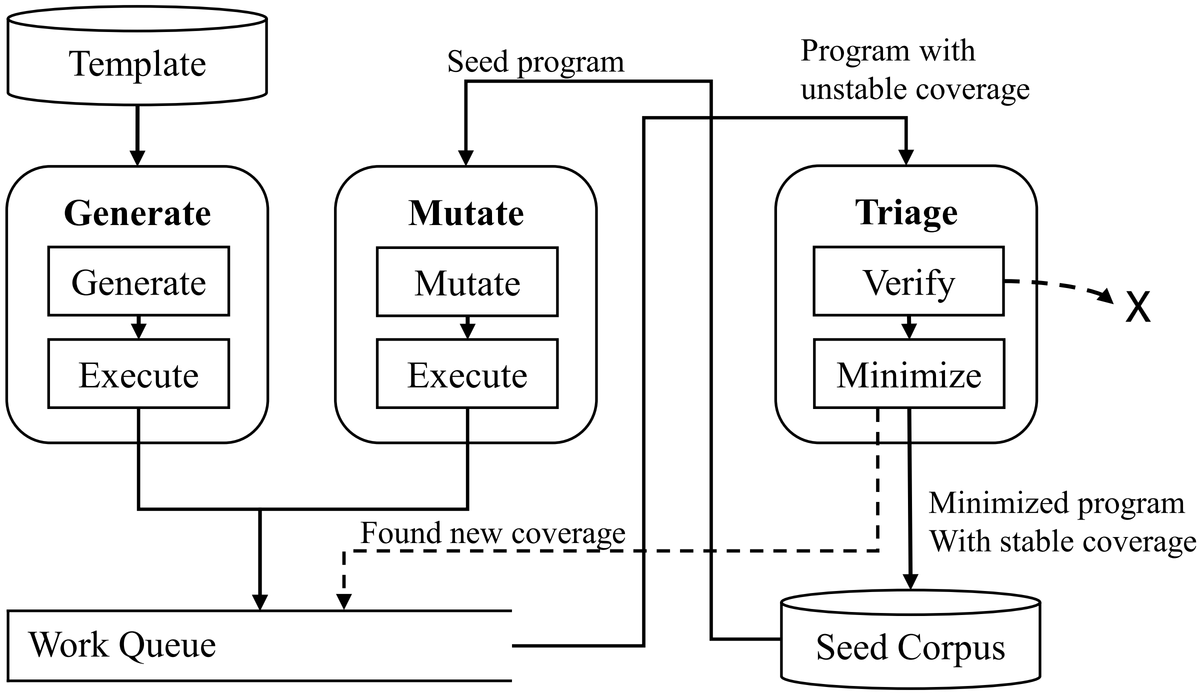
SyzVegas: Beating Kernel Fuzzing Odds With Reinforcement Learning

Daimeng Wang, Zheng Zhang, Hang Zhang
Zhiyun Qian, Srikanth V. Krishnamurthy, Nael Abu-Ghazaleh
University of California, Riverside

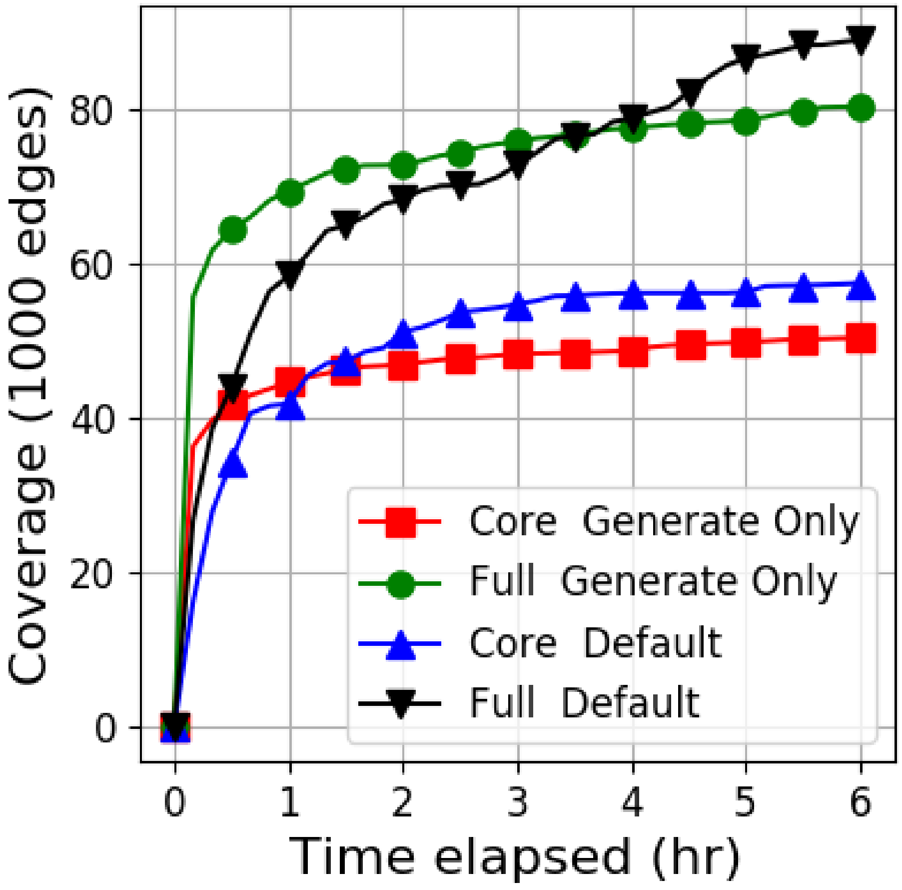
Introduction

- Fuzzing
  - Simple in principle
  - Difficult to optimize
- Kernel Fuzzing: Syzkaller
  - Evergrowing syscall templates
  - Numerous decisions based on
    - Strong intuitions and domain expertise
    - Empirical testing and tuning

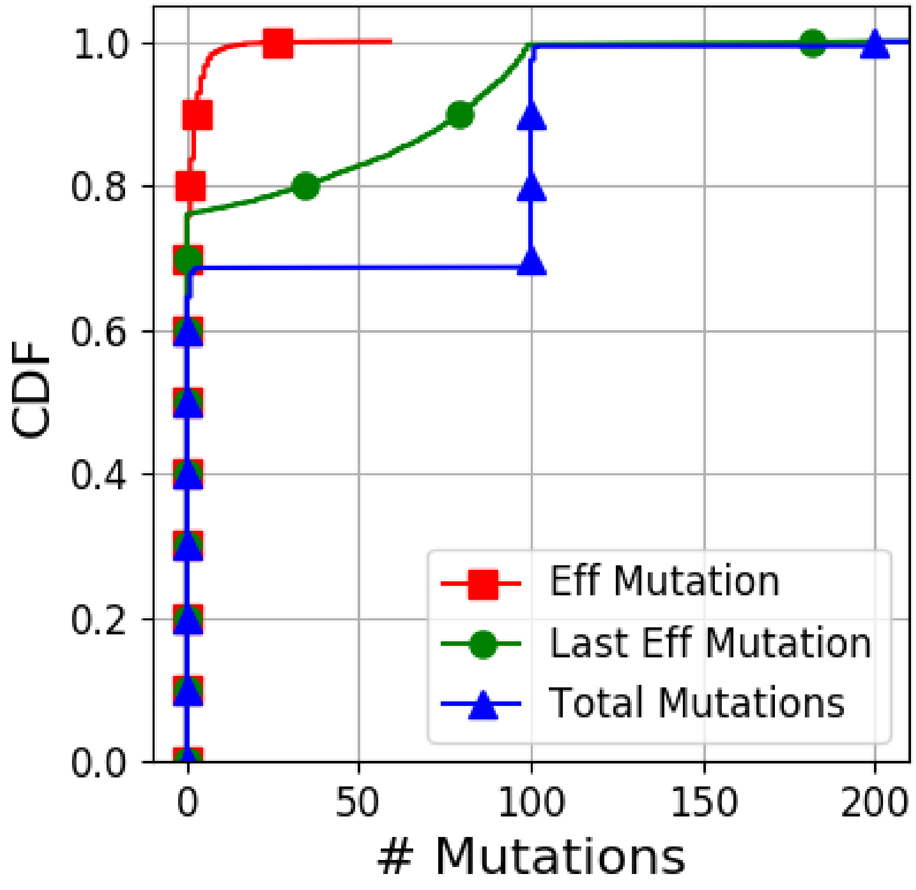


Observations

- Generation is powerful early-on

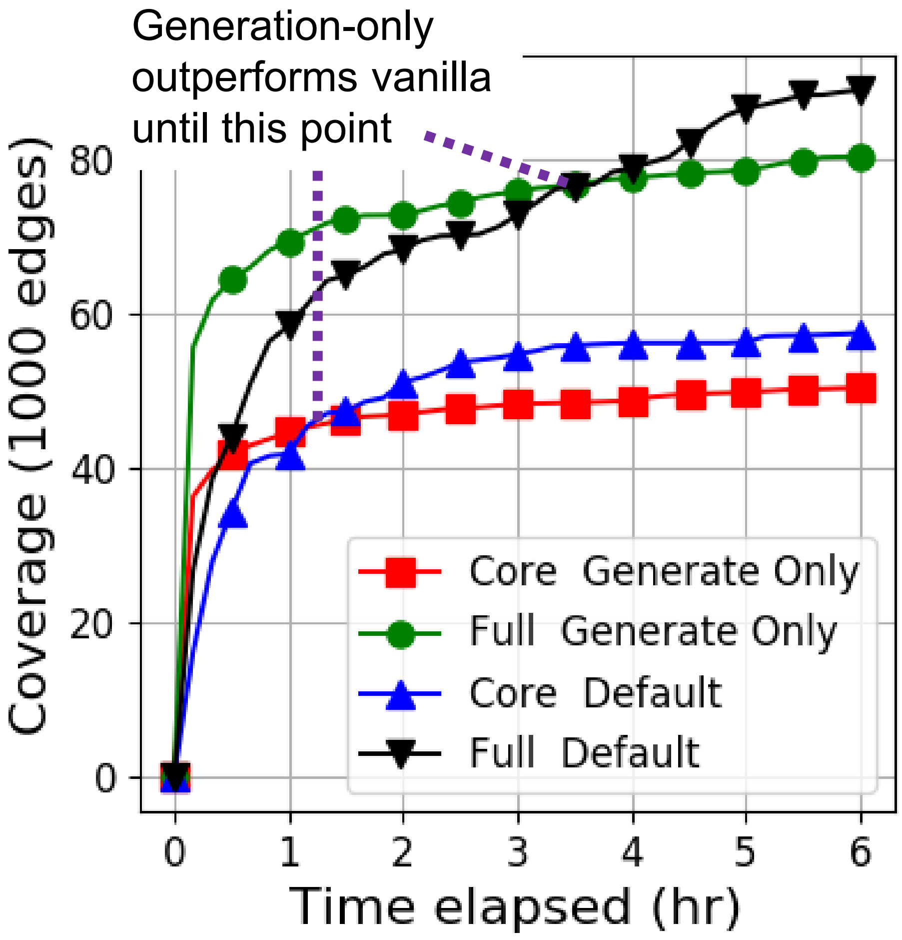


- Mutation can be more effective

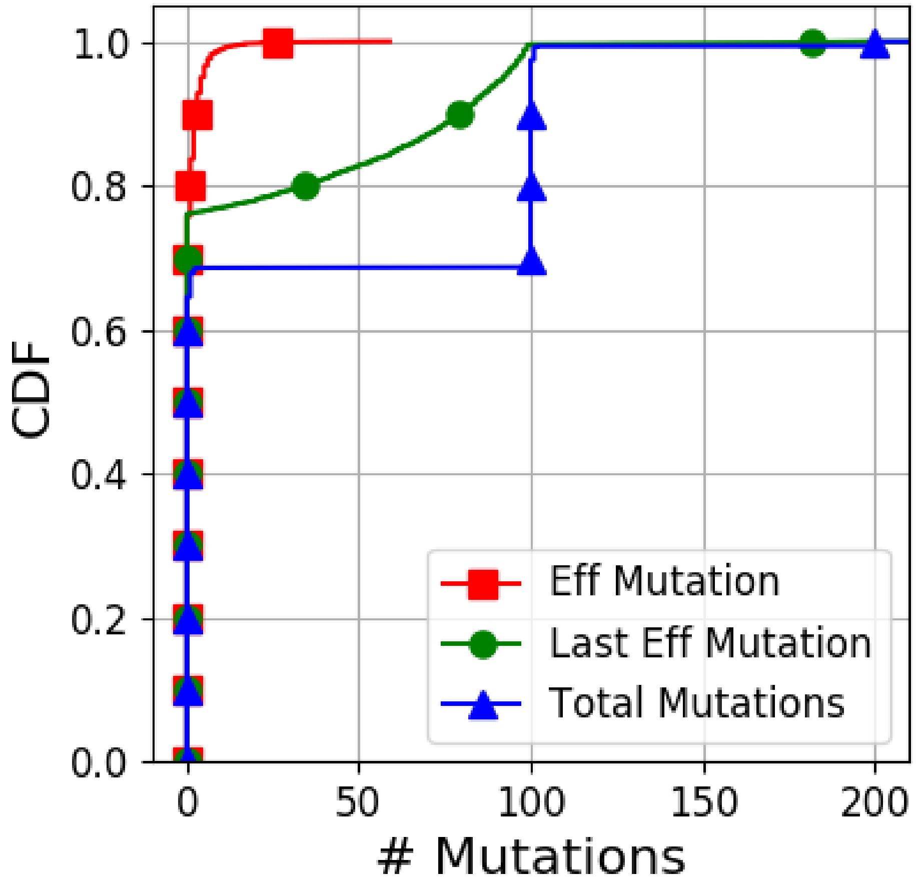


Observations

- Generation is powerful early-on

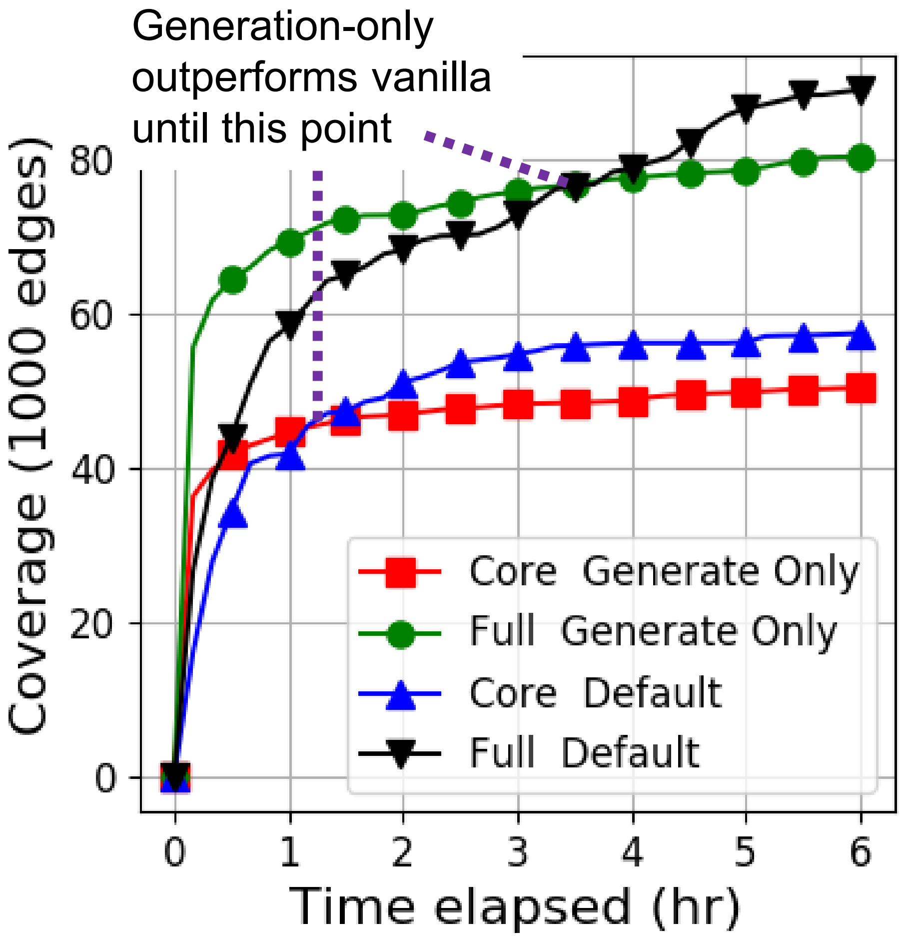


- Mutation can be more effective

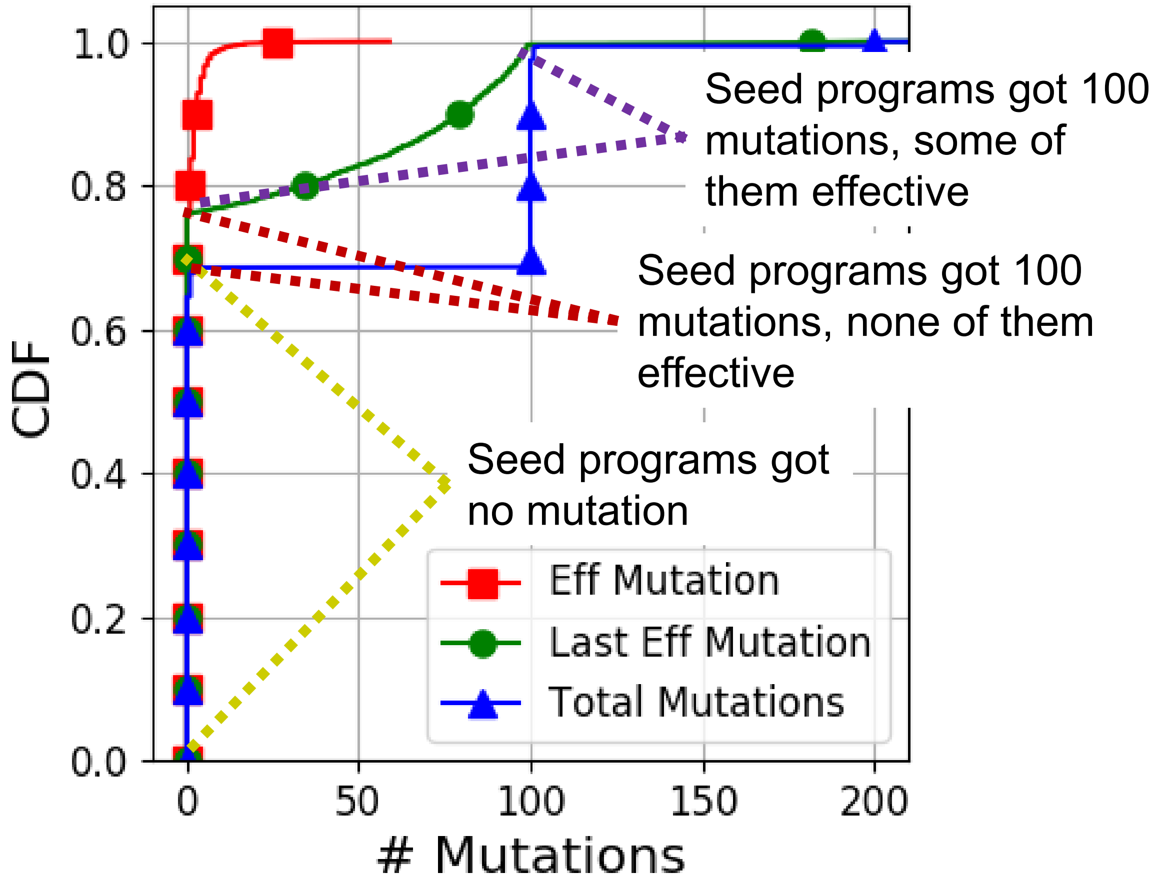


Observations

- Generation is powerful early-on



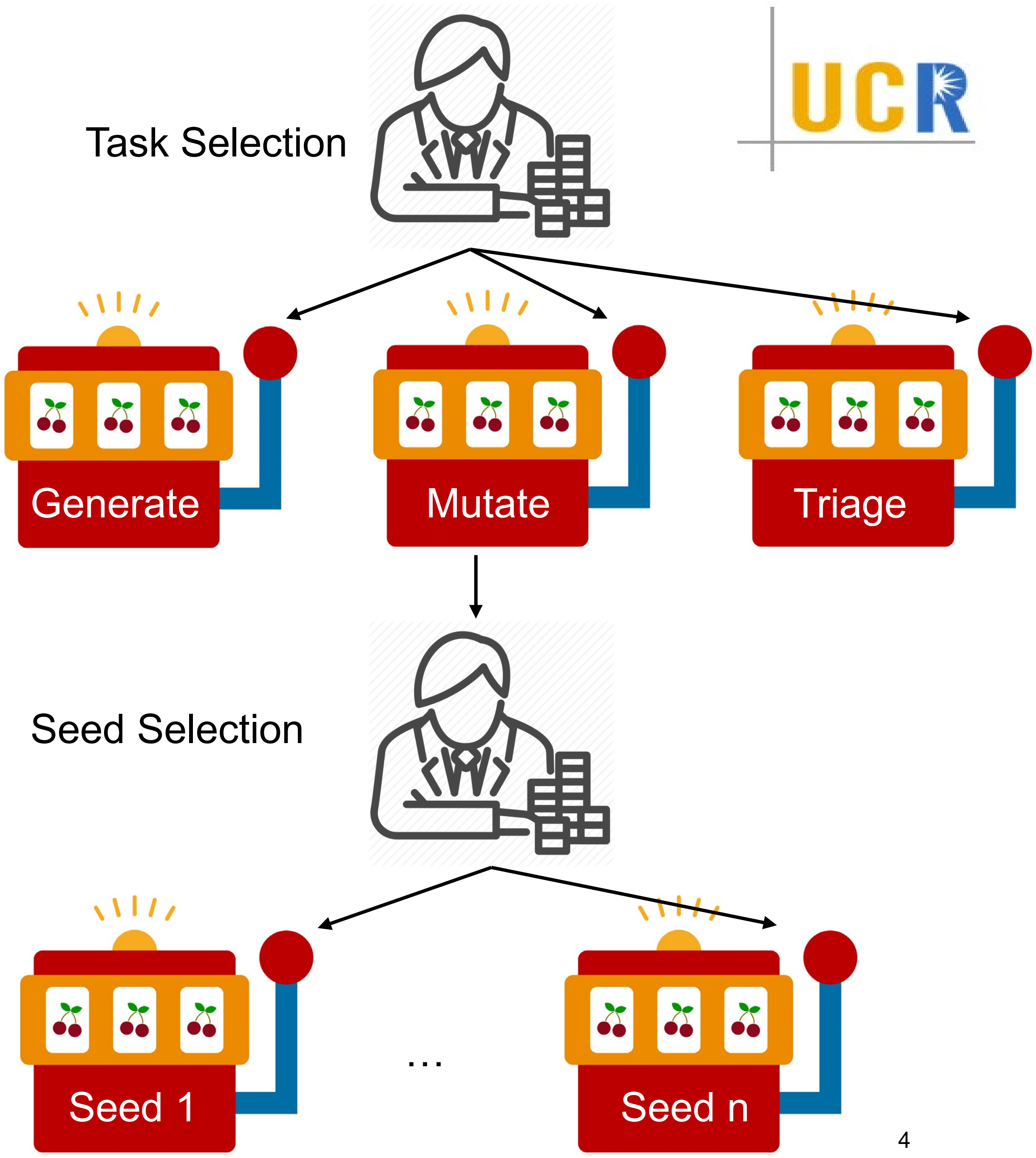
- Mutation can be more effective



Intuition

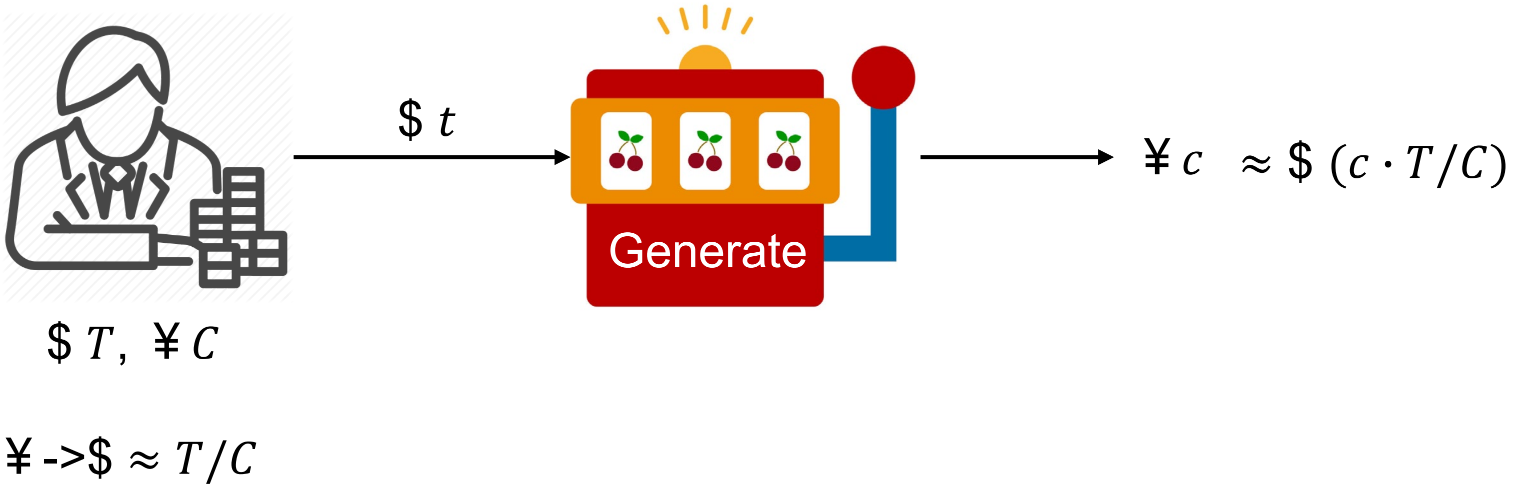
- SyzVegas: Treat task scheduling and seed selection as Adversarial Multi-armed-bandit (MAB) problems
  - MAB: No state required. Lightweight.
  - Adversarial: Reward of each choice changes over time.

- Challenge:
  - How to assess the reward?
    - Goal: Max coverage. Min time.
  - How to adapt
    - Goal: Fast reaction

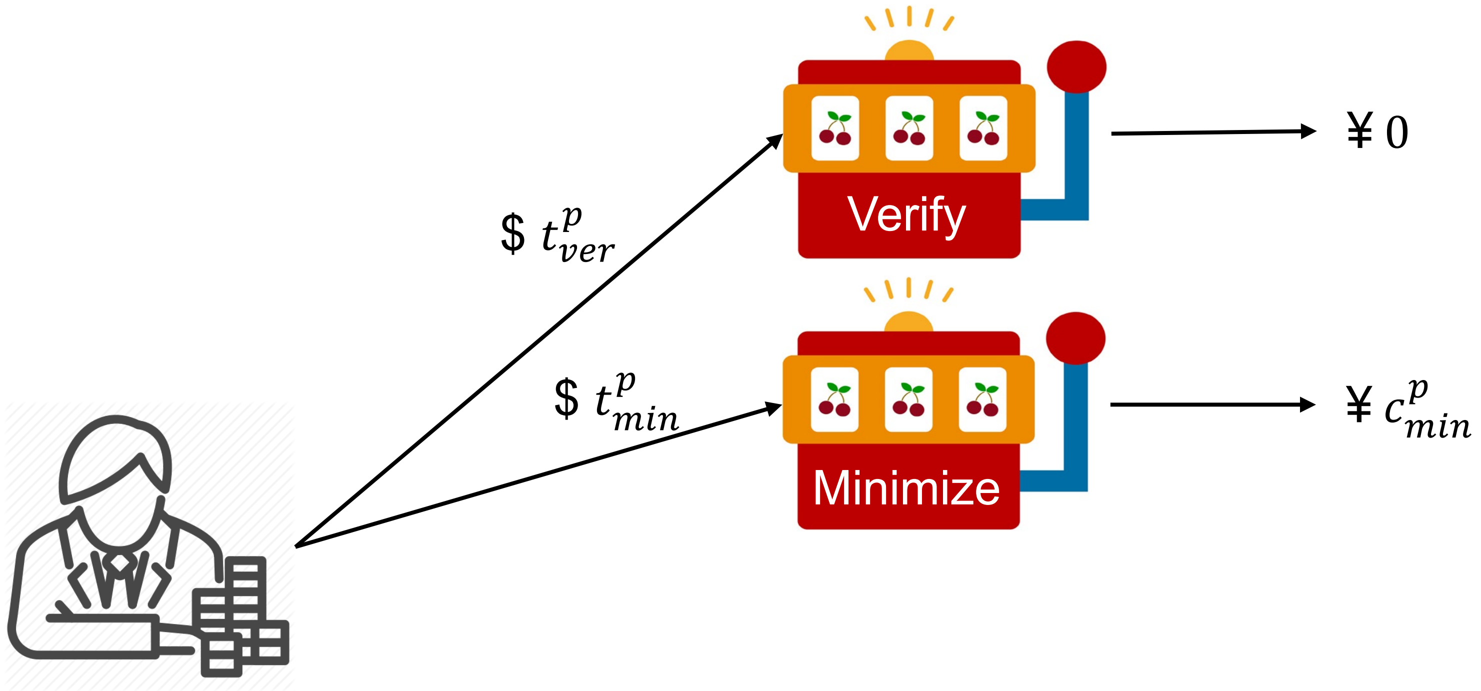


Reward Assessment

- Single reward metric that captures coverage and time spent
- Idea: Currency conversion
  - Conversion rate: total time / total coverage
  - $g = c \times (T/C) - t$



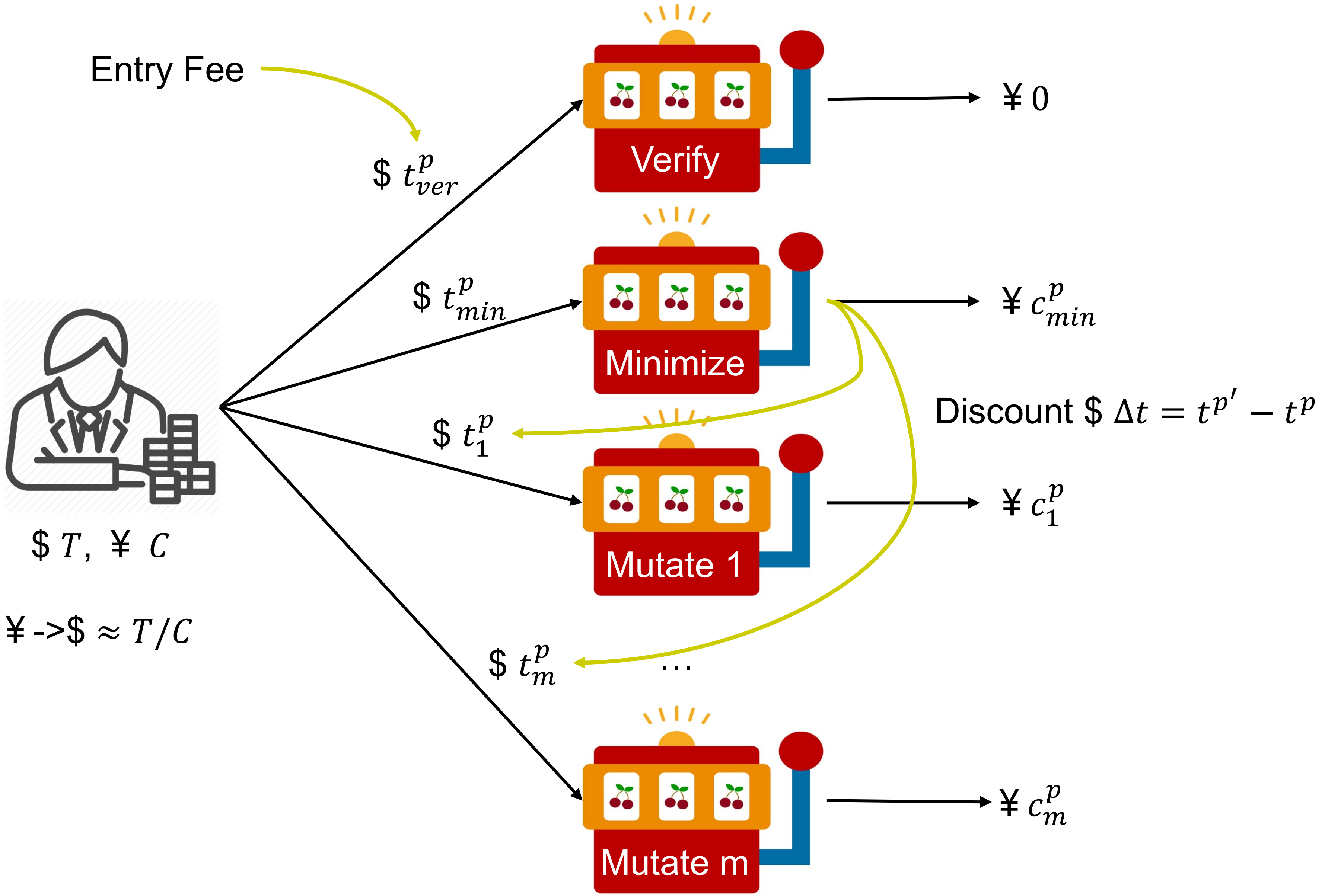
Reward Assessment: Triage, Mutate



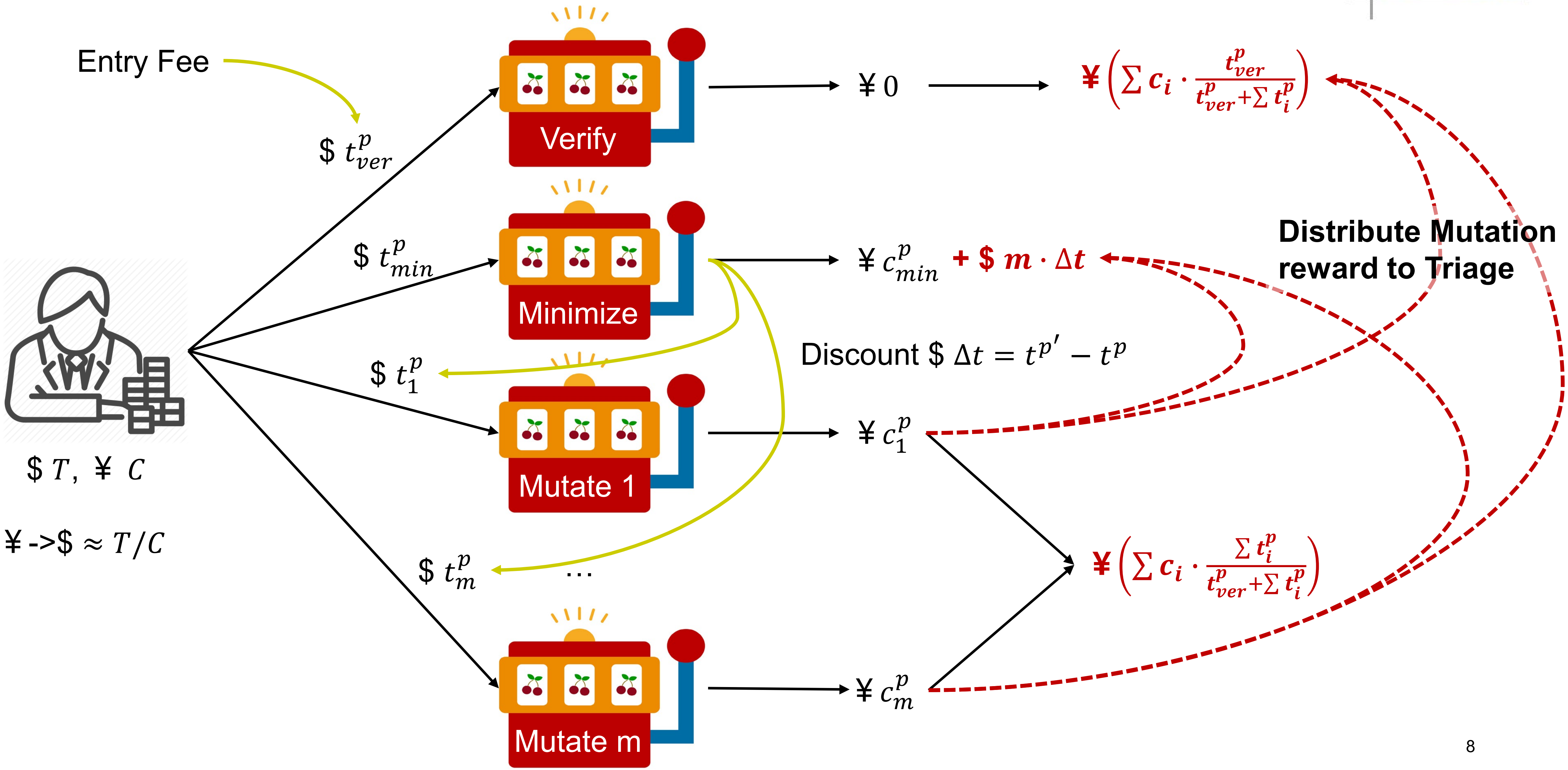
\$ T, ¥ C

¥ ->\$ $\approx T/C$

Reward Assessment: Triage, Mutate

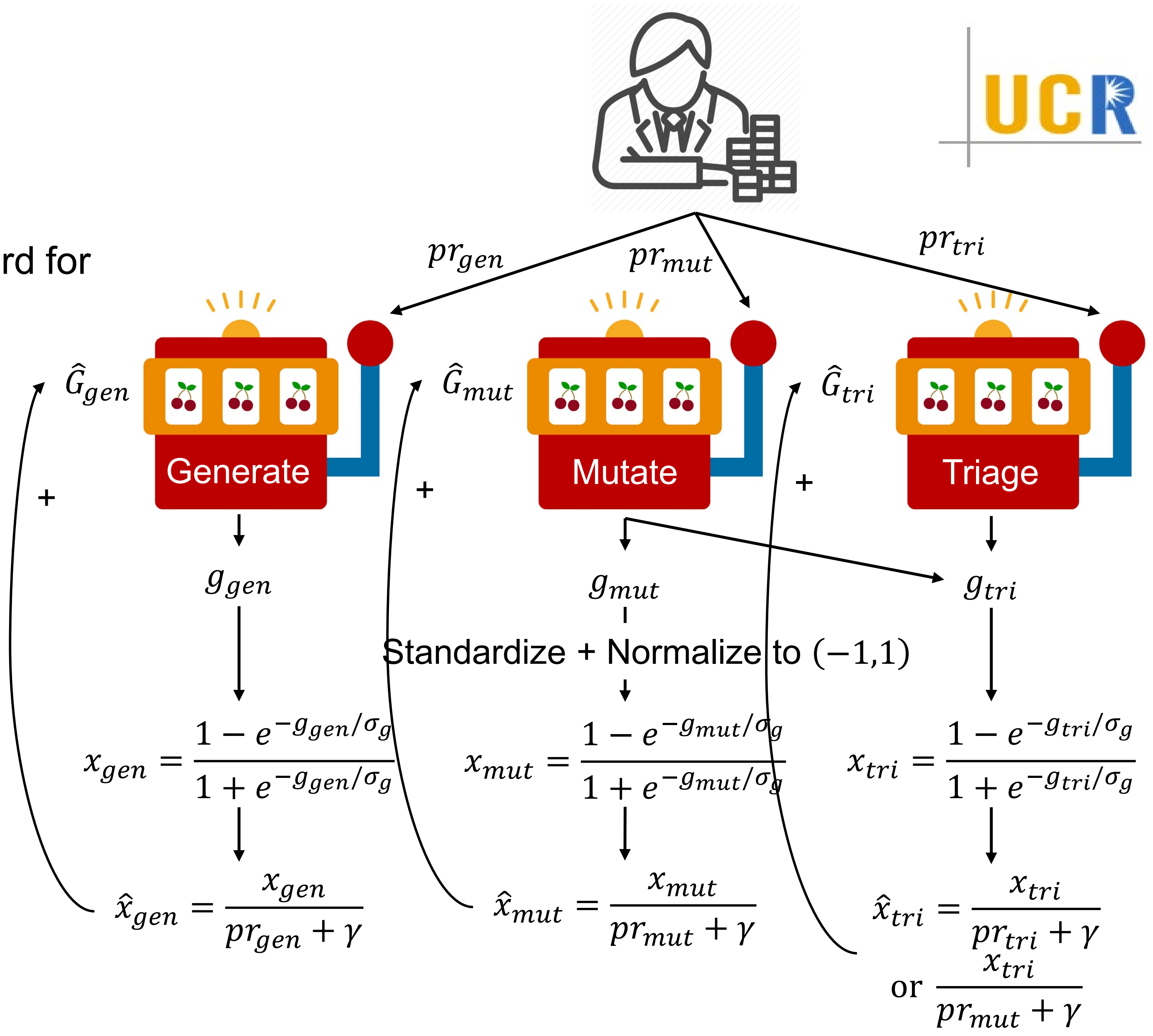


Reward Assessment: Triage, Mutate



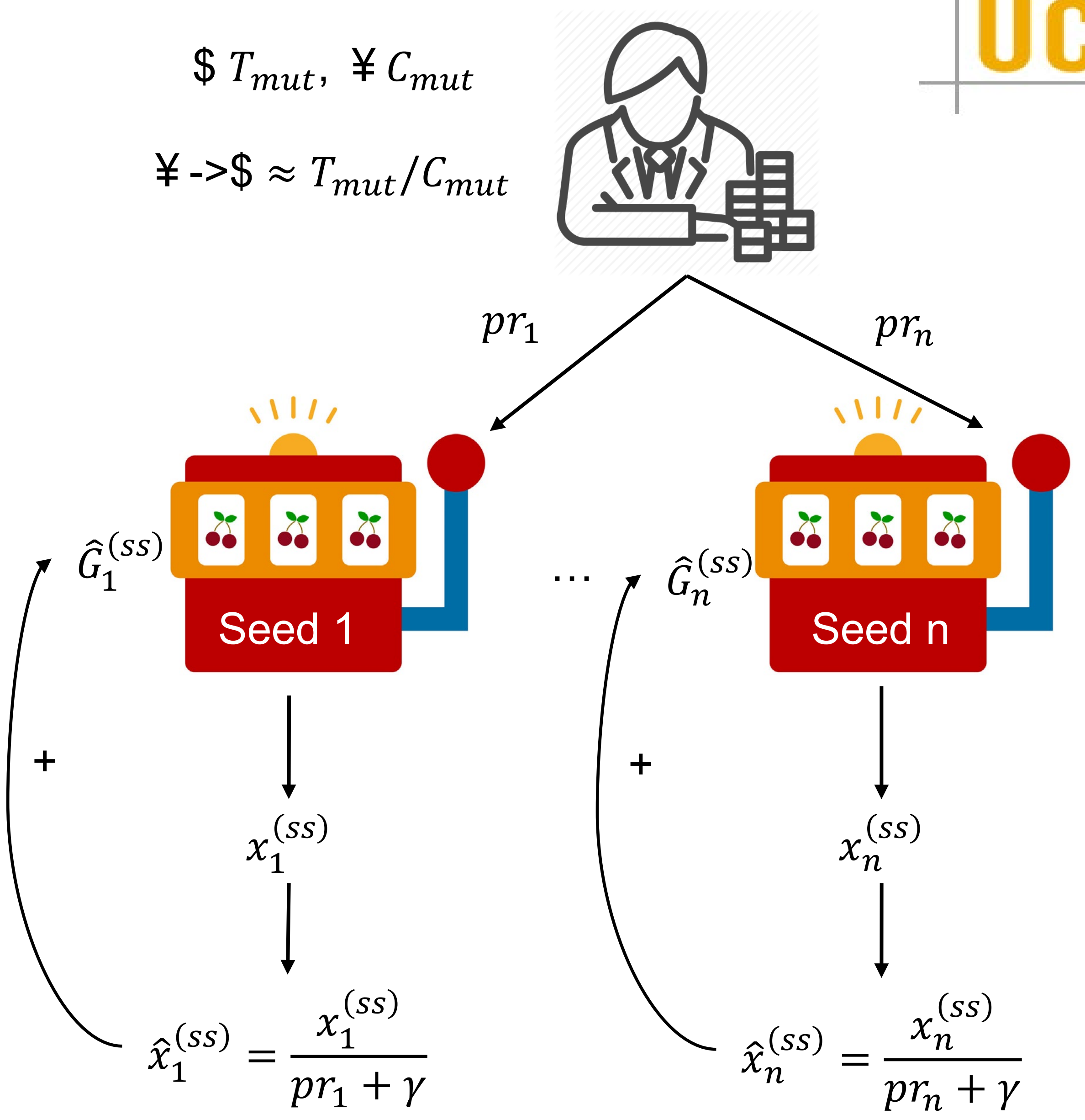
Task Selection

- Exp3-IX + Exp3.1
- Record total normalized reward for each task.
- After executing a task, adds the normalized reward to its total.
- Probability of selecting each task is exponential to its total reward.
- Reset the total reward to zero periodically.



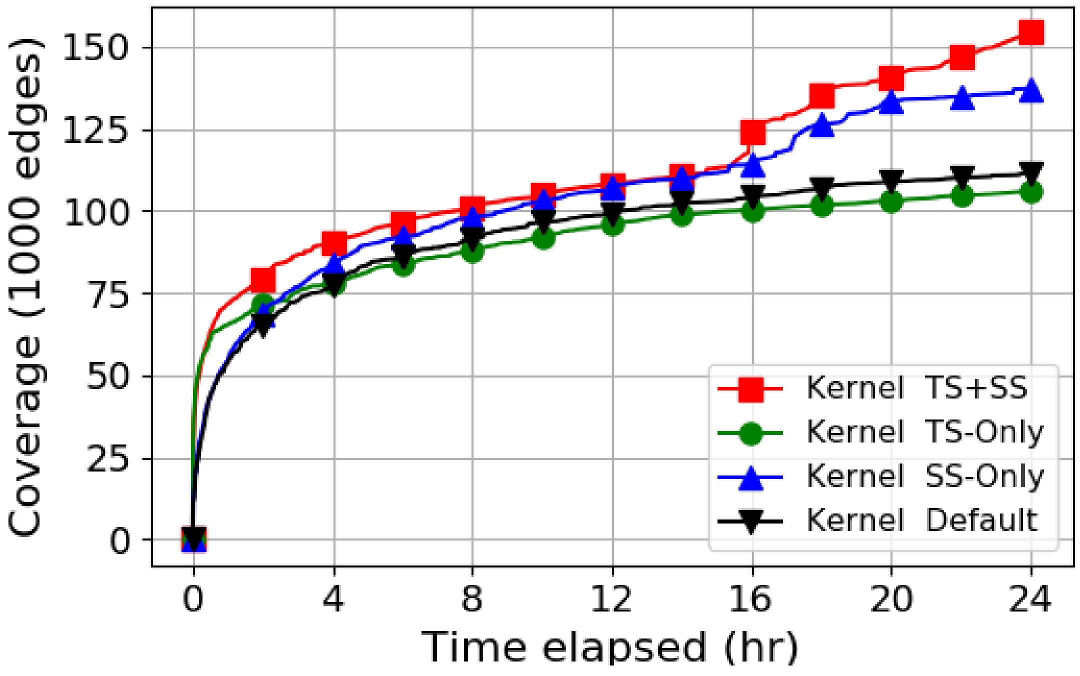
Seed Selection

- Different "conversion" rate
- No need to split reward with triage
- Ever-increasing number of arms
- No need to reset
  - Diminishing reward

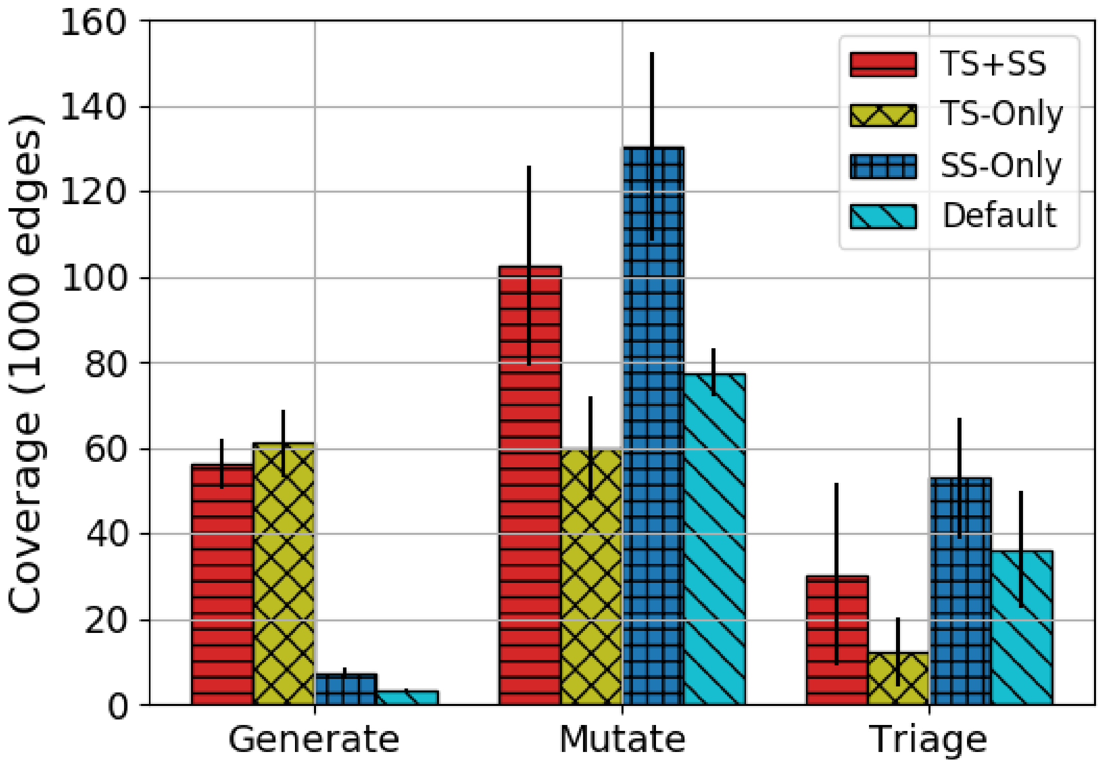


Evaluation: Linux Kernel 5.6.13

- Median Coverage
  - TS=Task Selection
  - SS=Seed Selection

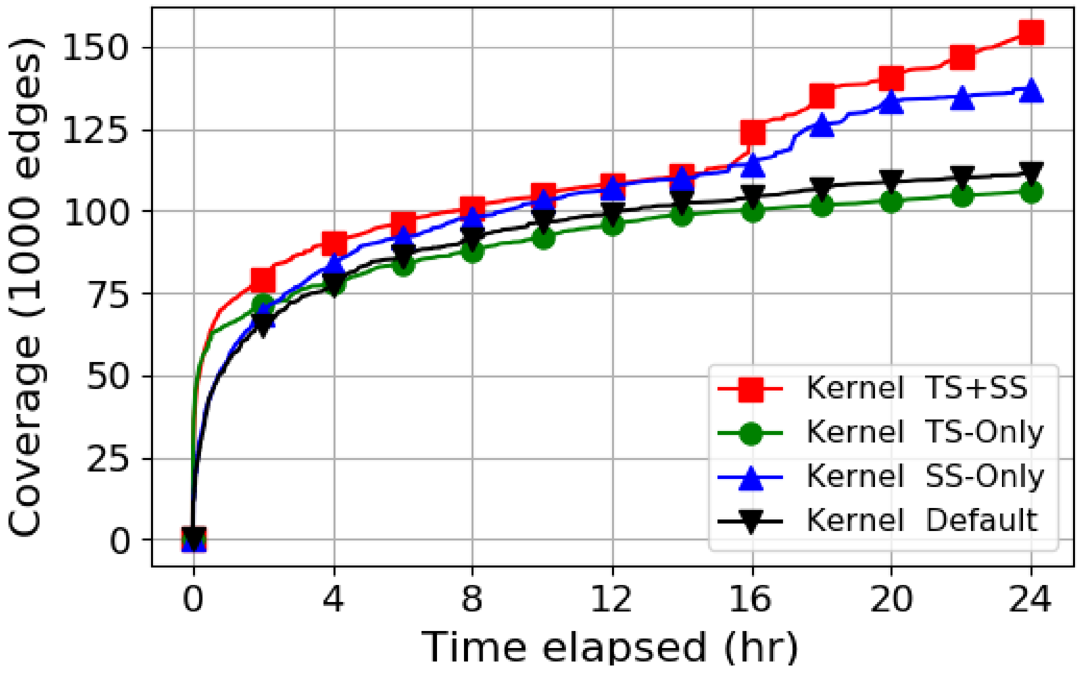


- Breakdown

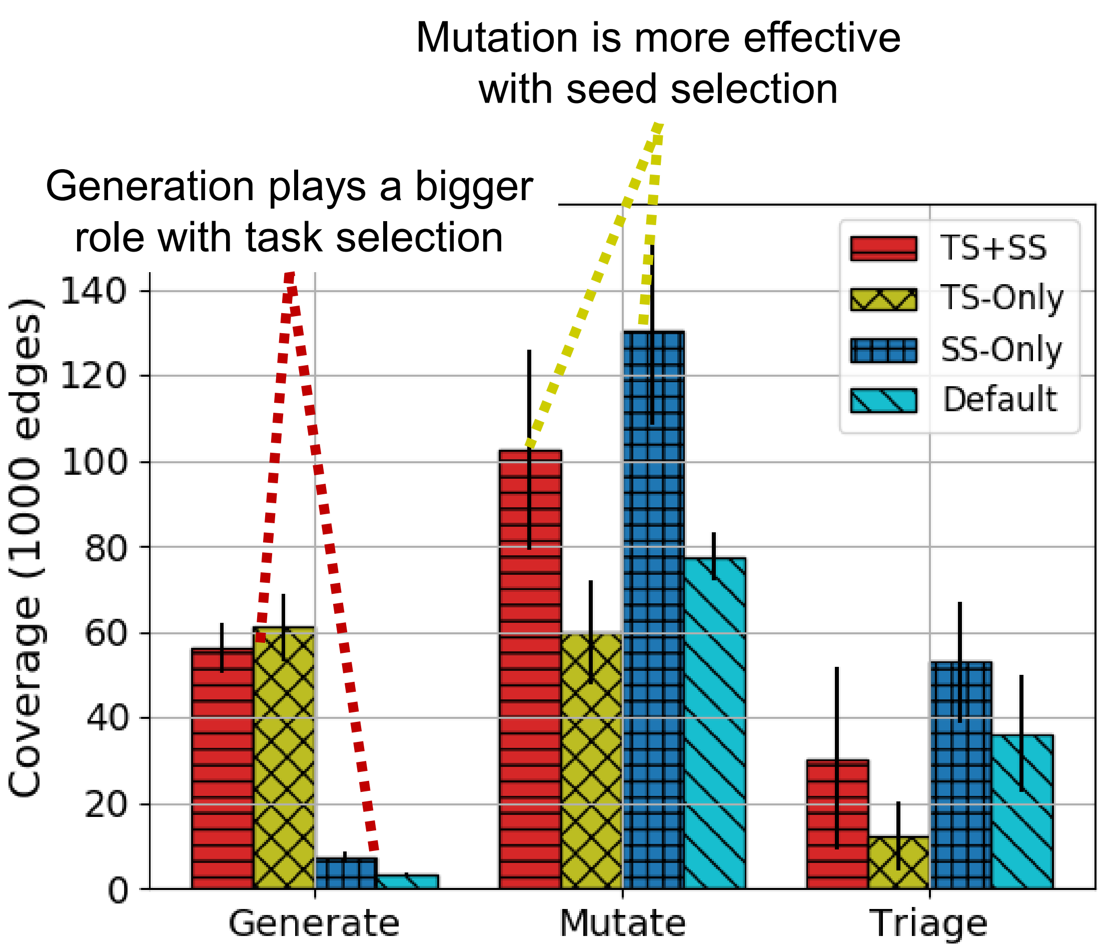


Evaluation: Linux Kernel 5.6.13

- Median Coverage
  - TS=Task Selection
  - SS=Seed Selection

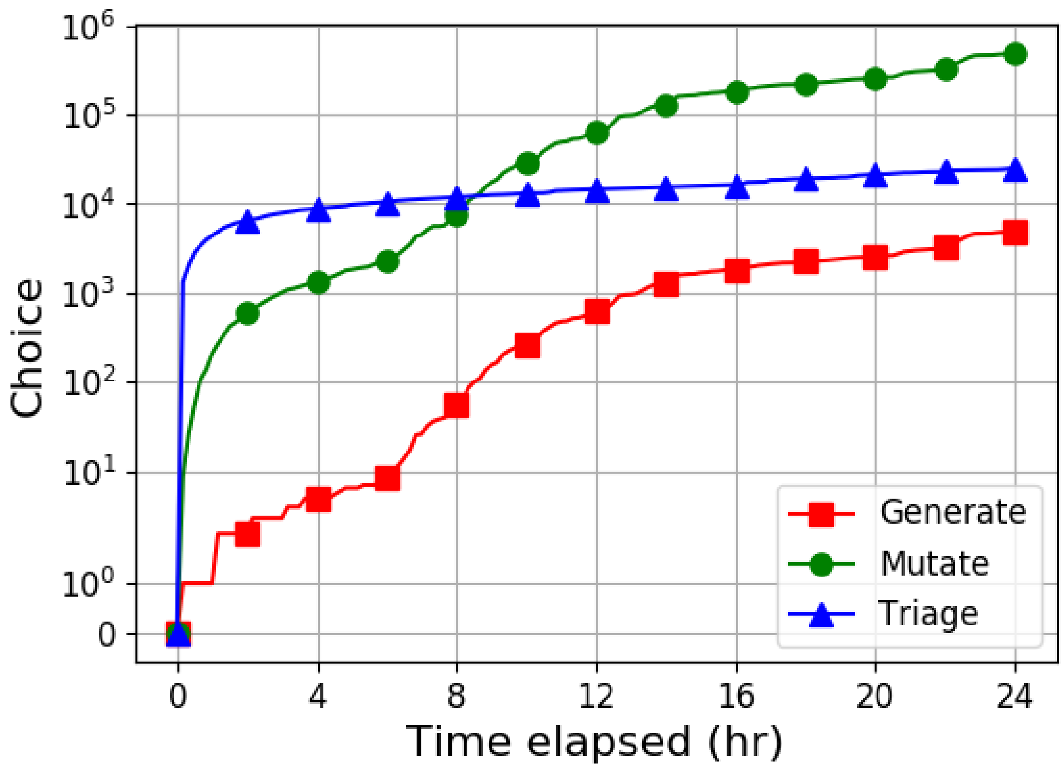


- Breakdown

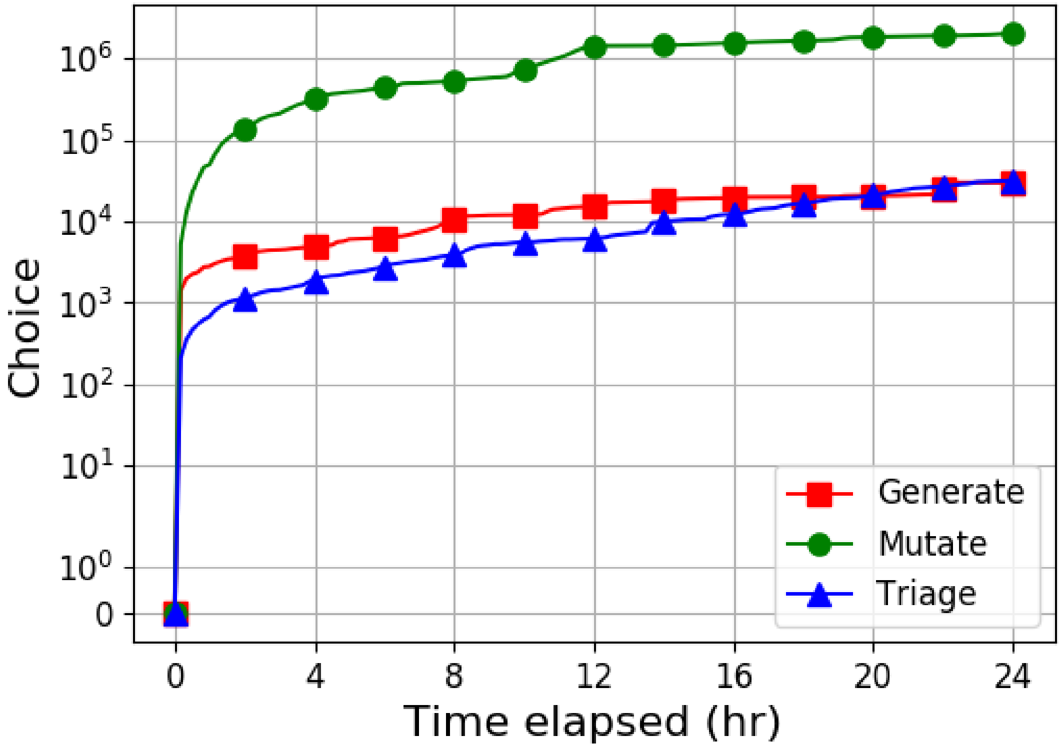


Evaluation: Task Choice

- Syzkaller

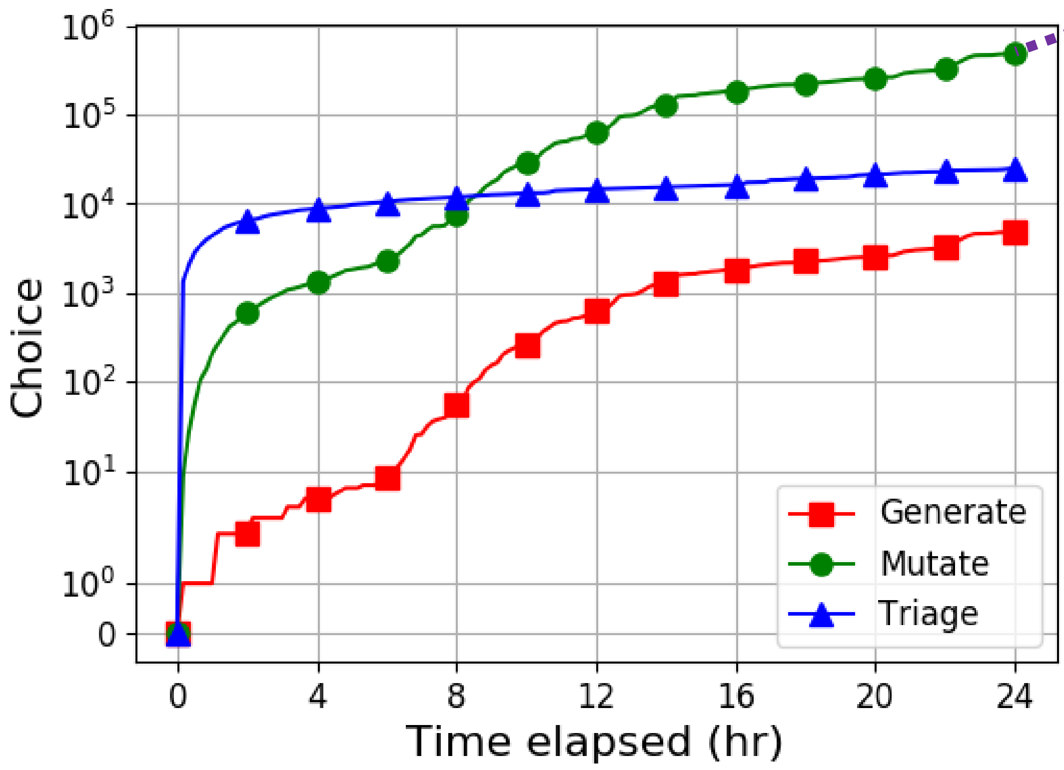


- SyzVegas



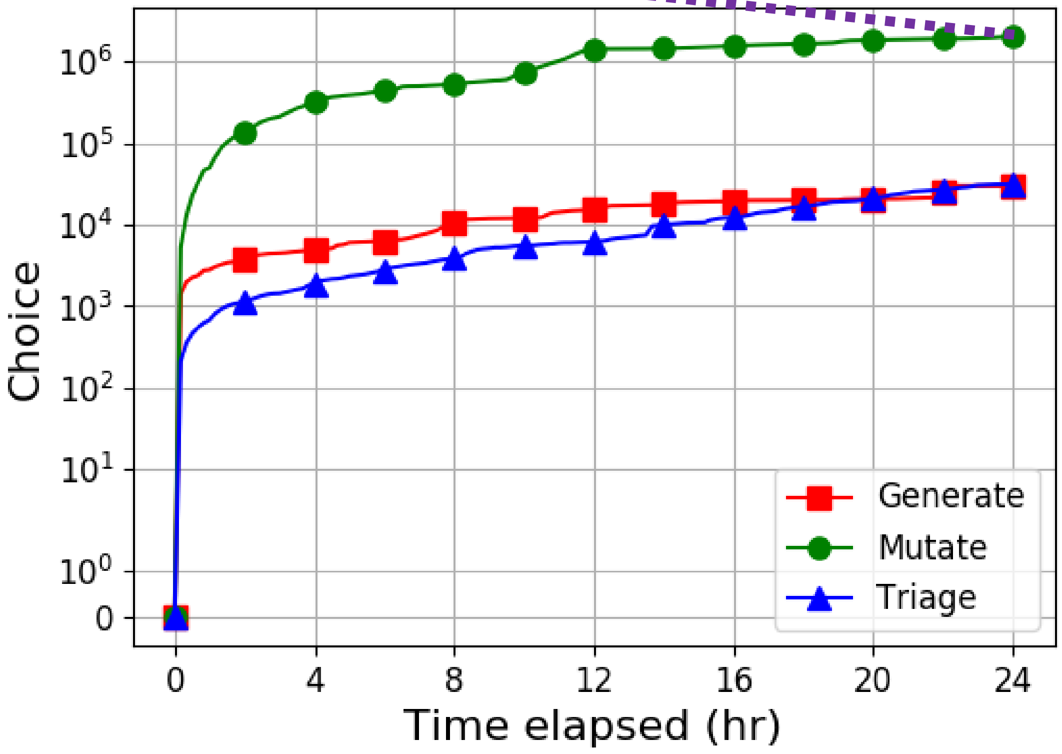
Evaluation: Task Choice

- Syzkaller



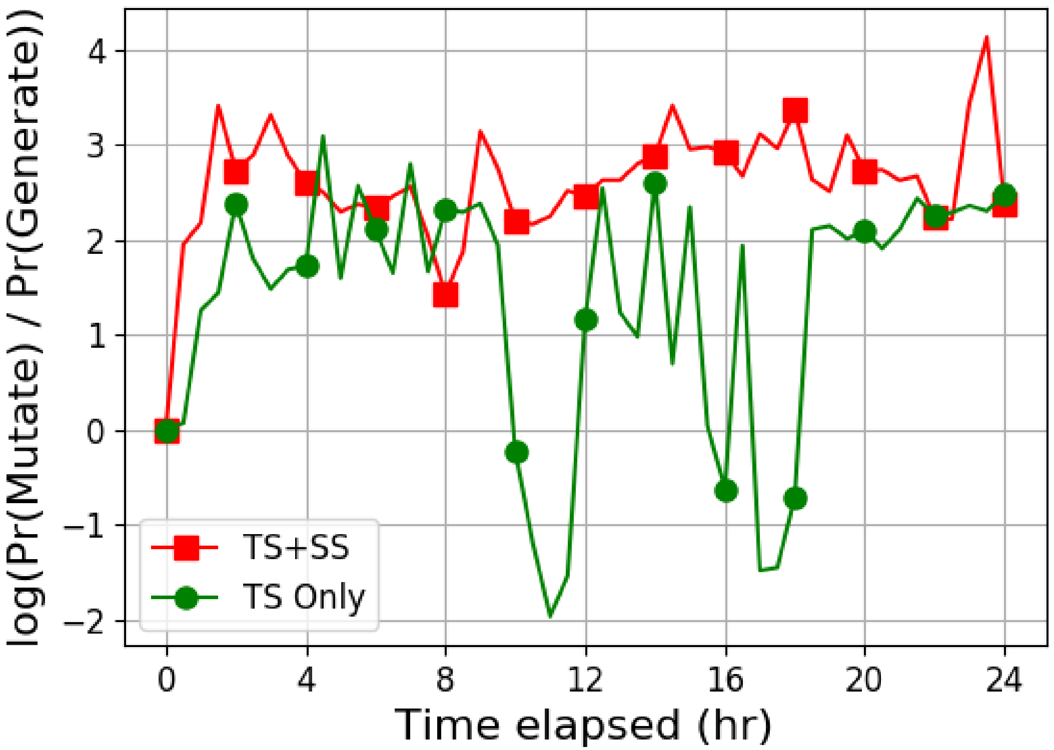
- SyzVegas

SyzVegas performs ~10 times more mutations than vanilla

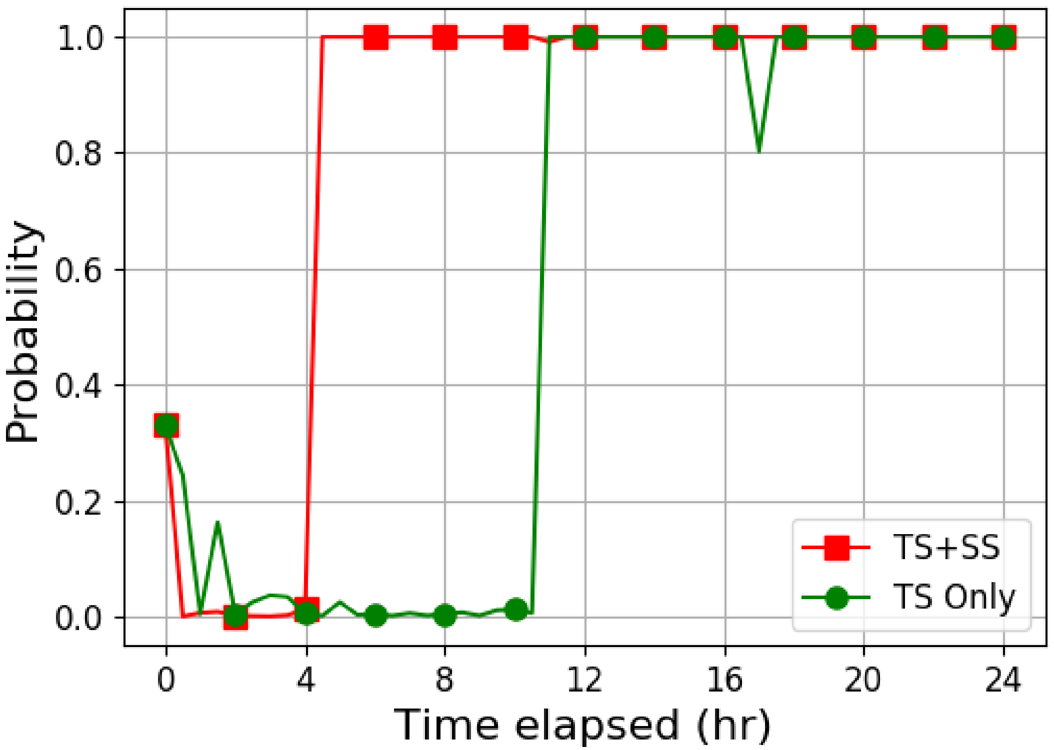


Evaluation: MAB Behavior

- Gen vs Mut



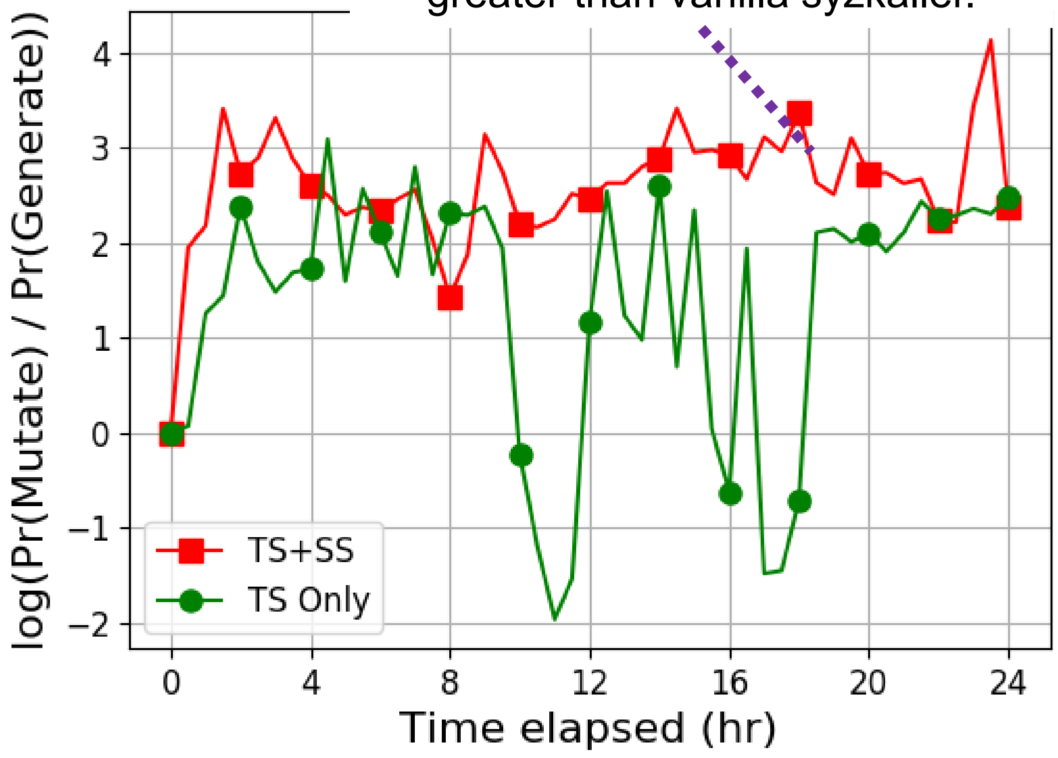
- Triage



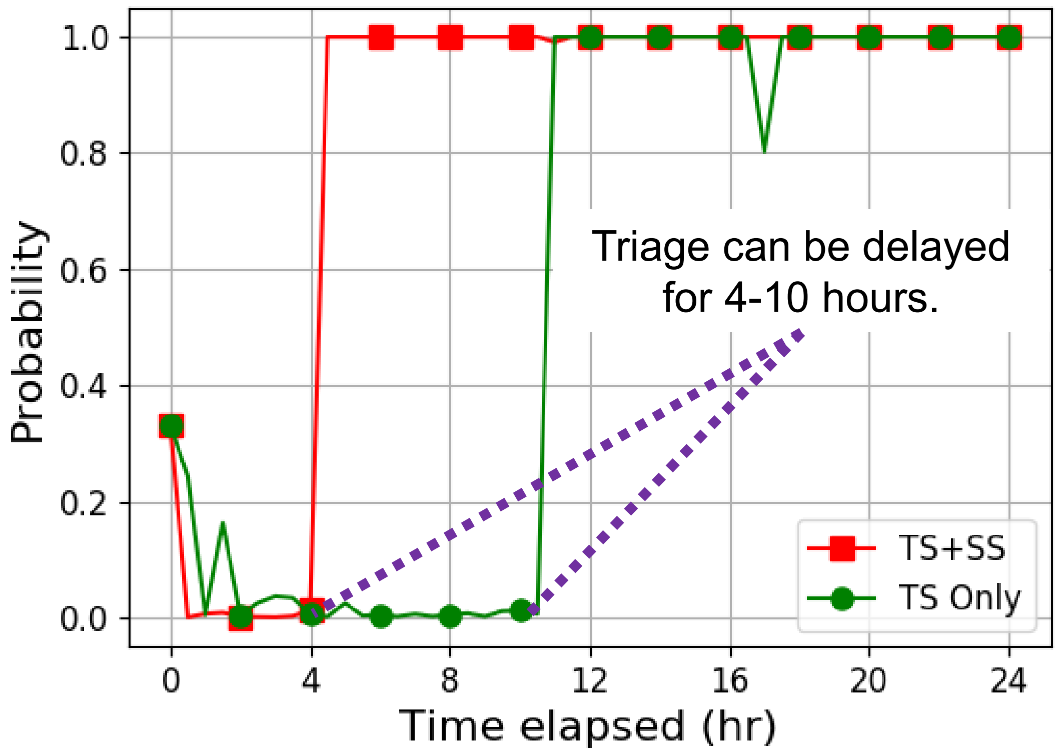
Evaluation: MAB Behavior

- Gen vs Mut

Mutation has 200-100 times probability than generation. This is greater than vanilla syzkaller.

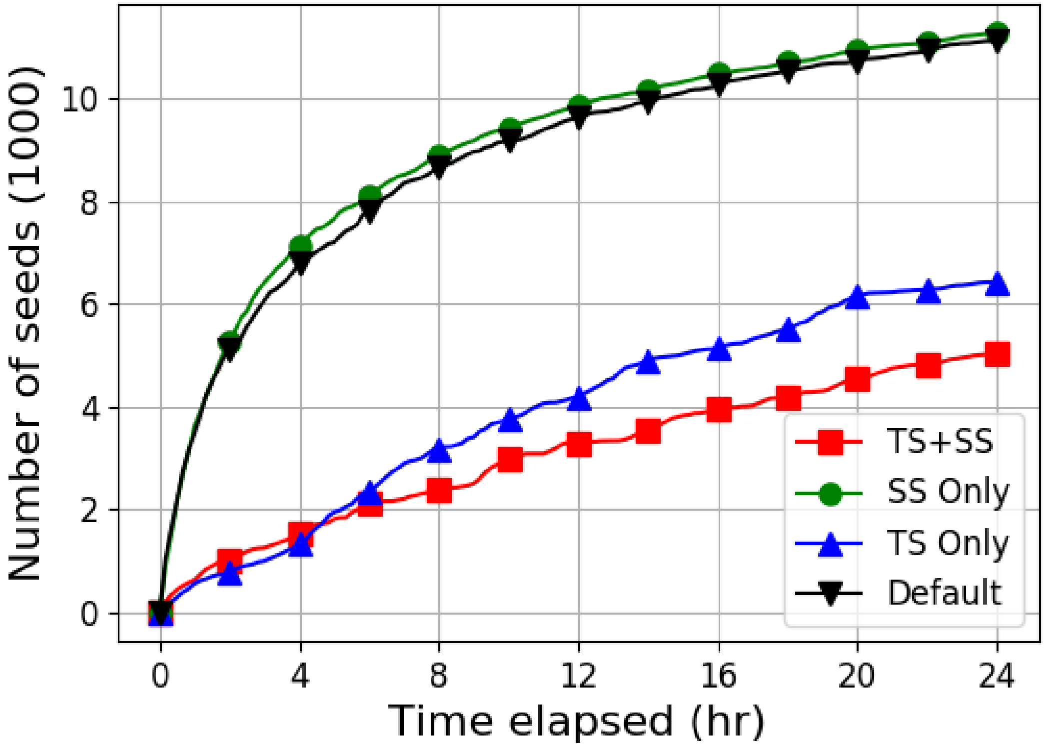


- Triage

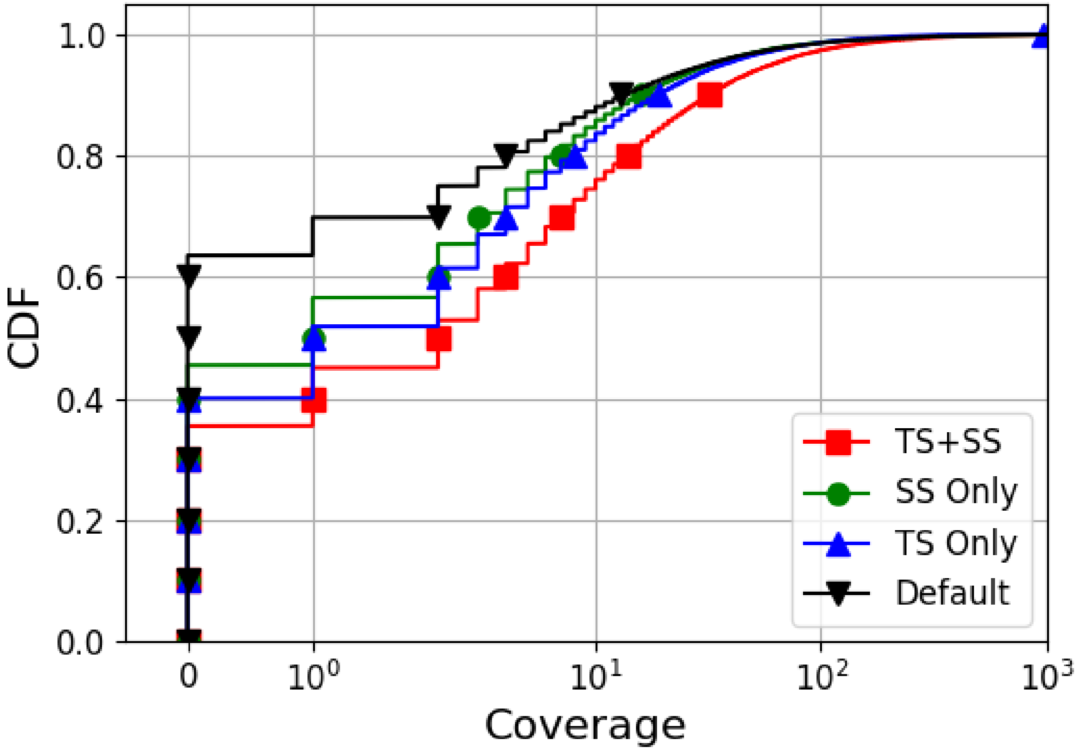


Evaluation: Seed Programs

- Seed Number

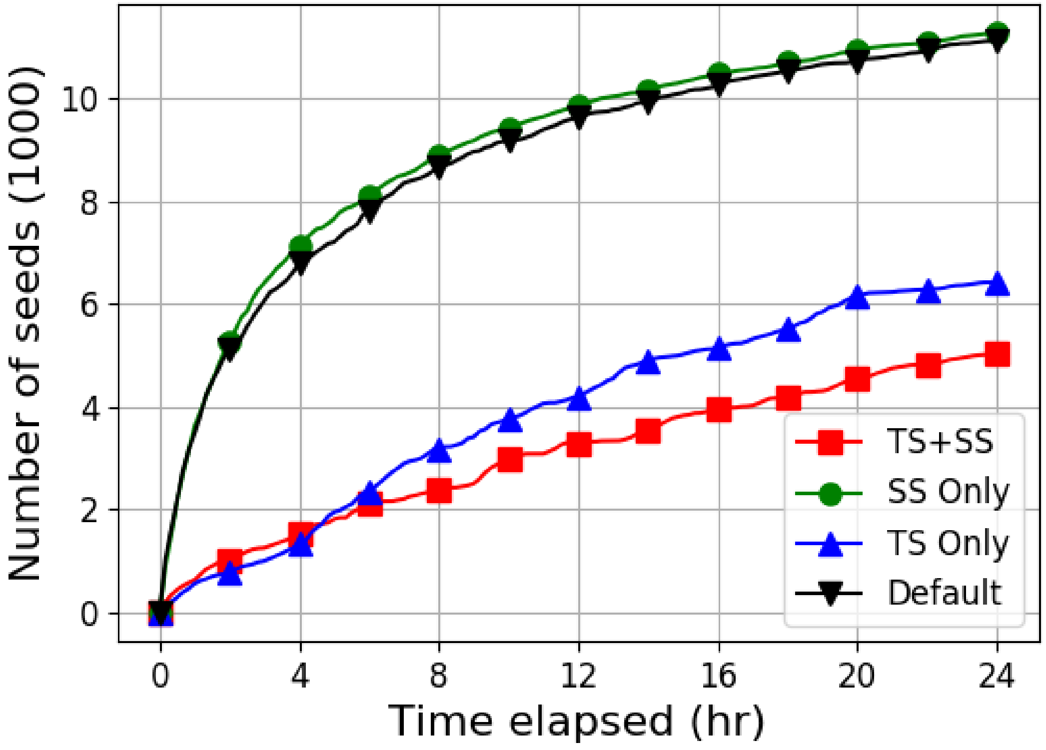


- Seed Power

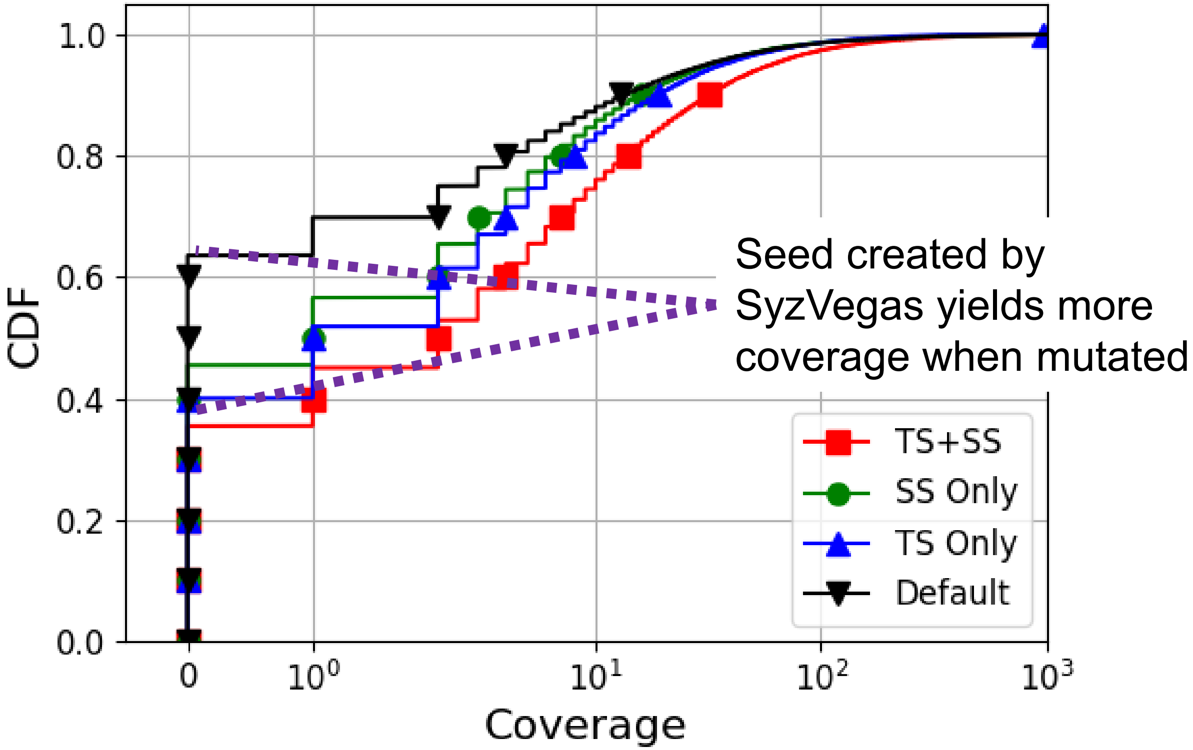


Evaluation: Seed Programs

- Seed Number

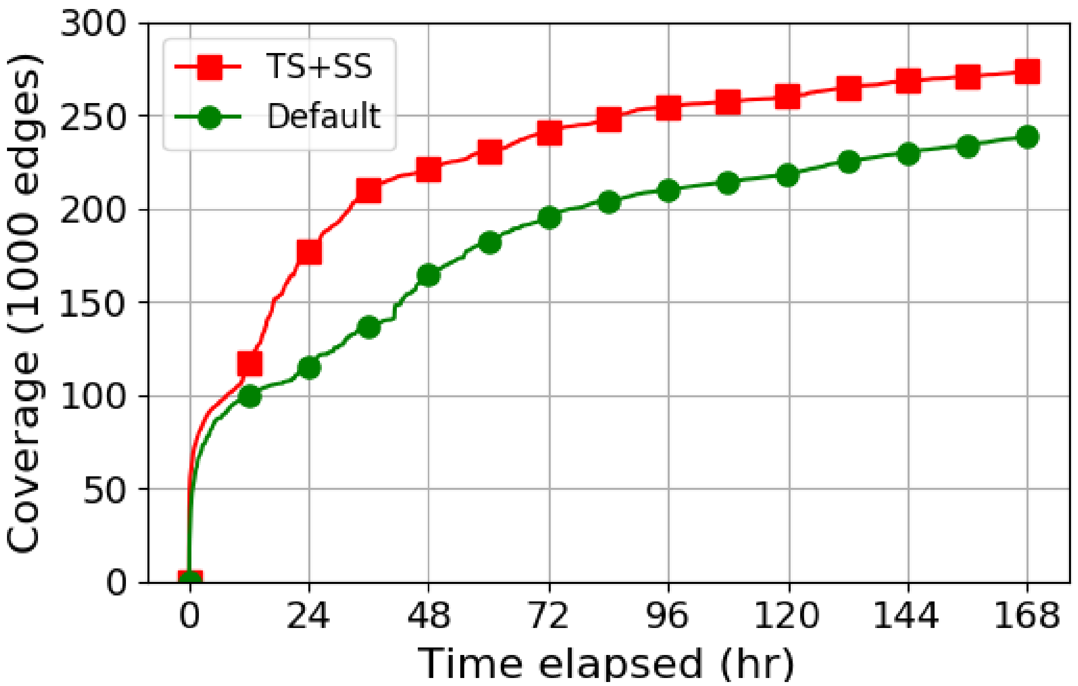


- Seed Power

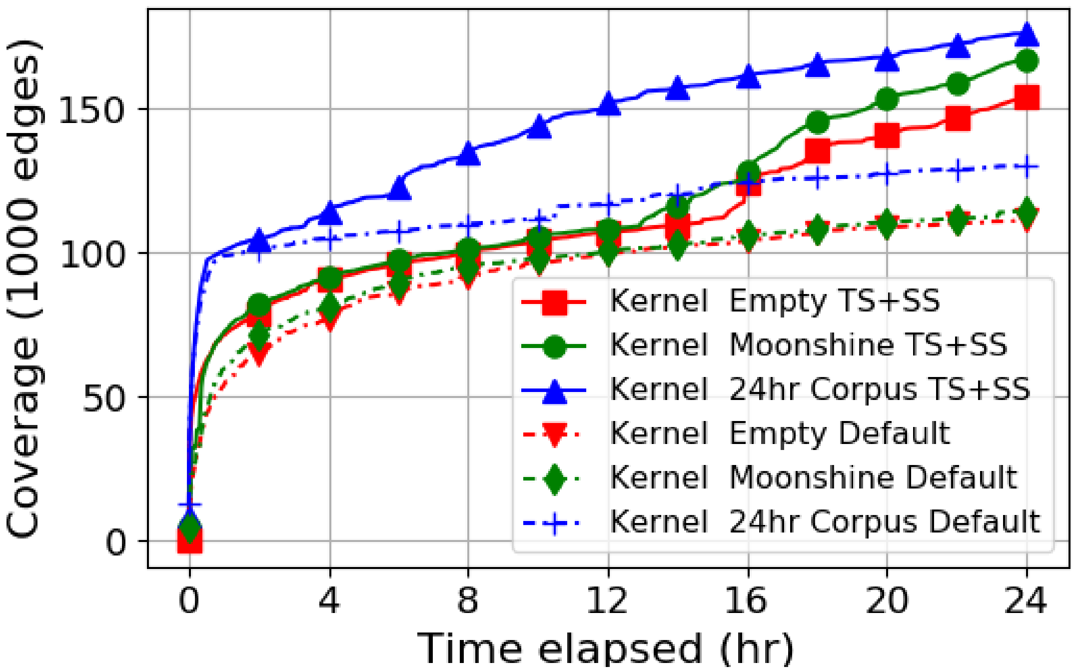


Evaluation

- 7 days



- With an initial seed corpus



For more experiments, check our paper ☺

Crash Type	Functions	Discovered		Status
		SyzVegas	Syzkaller	
Protection Fault	kmem_cache_alloc wait_consider_task	1	1	Already reported
RCU Stall	ext4_file_write_iter io_uring_release io_uring_setup tty_write	3	4	2 new and reproducible
Slab OOB	do_update_region vcs_scr_readw vgacon_scrolldelta vgacon_scroll	4	2	1 new but not reproducible
Use-after-free	ata_scsi_mode_select_xlat clear_buffer_attributes complement_pos con_scroll, do_update_region screen_glyph screen_glyph_unicode vc_do_resize vcs_scr_readw vc_uniscr_check vgacon_invert_region vgacon_scroll do_con_write	12	5	4 new but not reproducible
Warning	dev_watchdog generic_make_request_checks xfrm_policy_insert_list	3	3	Already reported

All crashes are fixed at the time of making this slide

Discussion & Future Work

- Adversarial MAB over other reinforcement learning
  - No definition of state required
  - Adapt to changing reward
  - Performance overhead 2.1%

- Combining with white-box methods
  - Static analysis, symbolic execution, etc.
  - Some Adversarial MAB algorithms (e.g. EXP4) can take external inputs

- Adjusting other parameters
  - Program size. Mutation operator choice, etc.
  - https://github.com/google/syzkaller/issues/1950

Conclusion

- Identify opportunities of optimization
- Introduce dynamic fuzzing to Syzkaller
- Improve coverage growth

- Git Repo:
  - https://github.com/seclab-ucr/SyzVegas
- Upstream effort:
  - https://github.com/google/syzkaller/pull/1895

Thanks!

Q & A Contact:

- Daimeng Wang: *dwang030@ucr.edu*
- Zhiyun Qian: *zhiyunq@ucr.edu*