Automated, Cost-Aware FaaS in Heterogeneous Environments

Matt Baughman

Parsl & funcX Fest
October 28, 2021

The State of Heterogeneity…

	HPC	Cloud	Edge
Description	• System-level homogeneity • Enterprise parts • Increasingly accelerators	• Configurable • Enterprise parts • Extensive resources • accelerators	• Integrated systems • Varying infrastructure • Low-power, consumer components
Related Metrics (time and…)	• Allocation units • Queueing overheads	• Allocation units • $$$ (money)	• Resource capability • Overhead costs
Current Examples	• DOE Computing Infrastructure • University Clusters	• AWS • Google Cloud • Chameleon	• Raspberry Pi • Amazon Greengrass • Particle Cloud

The State of Heterogeneity… Applications

- Instrument Control
  - Real-time
  - Time-sensitive
  - Compute offloading
- HPC workloads
  - Configuration dependent
  - Resource selection
  - Deployment and scaling
- Edge and sensor networks
  - Wide ranging information
  - Low power → needs execution environment
- Machine Learning/ Artificial Intelligence
  - Utilization of accelerators/ ASICs
  - Data and location dependent

Serverless enables compute anywhere…

- An ever growing number of tasks…
- An ever growing number of resources to use…
- How do we enable effective distribution?

```
def control_stuff(time_sensitive_data):
    return(doing_stuff)
```

```
def definitely_recursion(recursive_data):
    return(definitely_recursion(recursive_data))
```

```
def sort_stuff(unsortable_stuff):
    return(stuff_sorted)
```

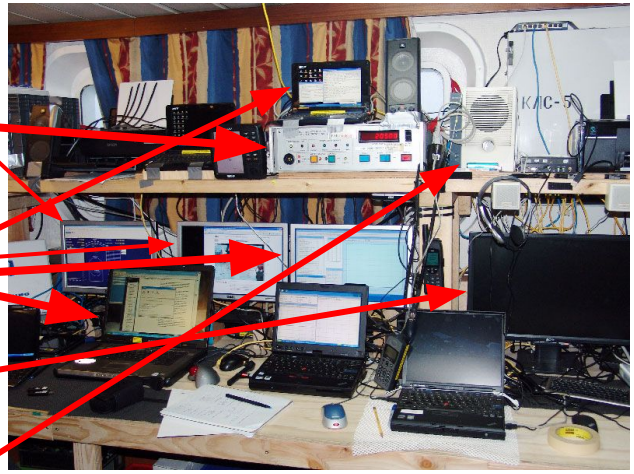
```
def infinite_loop(True):
    while True:
        1+1
    return(unlikely)
```

```
def do_stuff(big_data):
    return(stuff_done)
```

```
def solve_collatz(start_value):
    if start_value>0 and start_value <= np.inf:
        return(1)
    else:
        return(1)
```

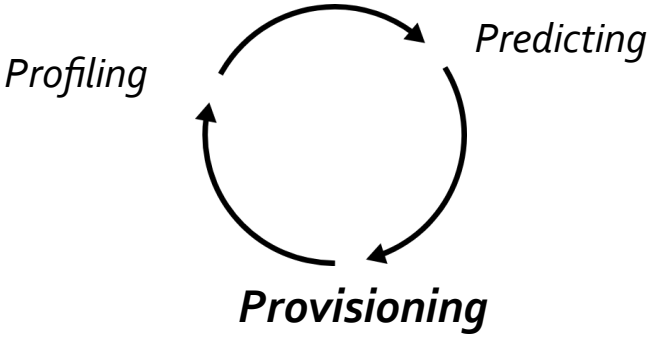
```
def parallel_stuff(1):
    return([1]*np.inf)
```

```
def p_equals_np(math):
    return(True)
```



(“modern” distributed system)

Cost-Aware Execution in Automated Serverless



Delta+ Overview

- Automated coordination of FaaS ecosystems
  - Built on top of funcX
- Unifying compute across the edge, HPC, and cloud
- Design components
  - Remove endpoint selection from end-users
  - Allow for tunable optimization metrics
  - Enable intelligent use of all available resources
  - Handle dynamic ecosystems
  - Dynamic resource provisioning
  - Multidimensional cost and constraint satisfaction

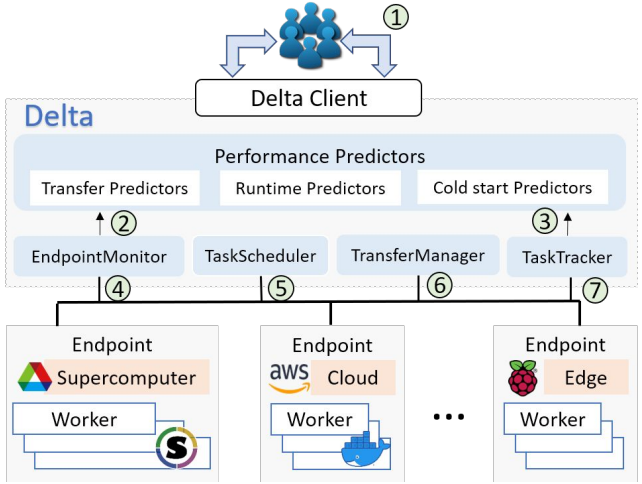
```python
from delta import DeltaClient

# Define function to be scheduled via Delta
def classify(img):
    import tensorflow as tf
    ...
    return model.predict(img)

# Create client and register function
client = DeltaClient()
func_id = client.register_function(classify)

# Request task to run function on image
img = ...
task_id = client.run(img, function_id=func_id)

# Retrieve task result
res = client.get_result(task_id, block=True)
```



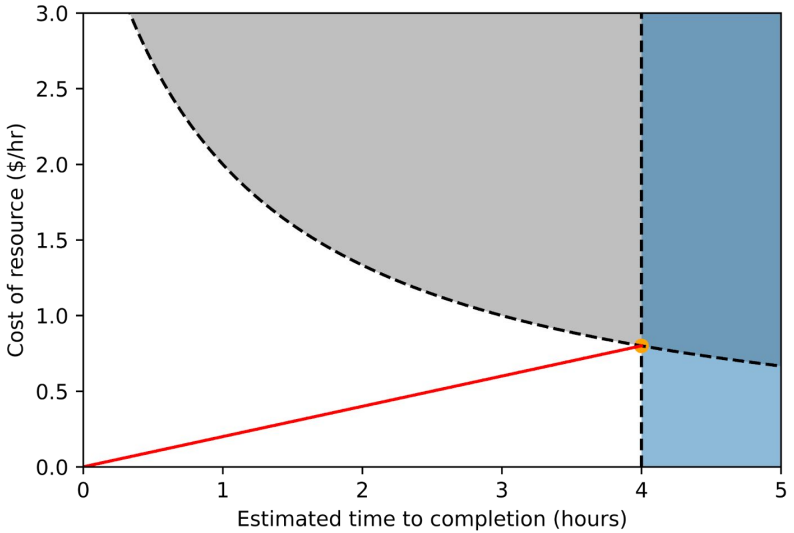
Kumar, Baughman, et al. "Coding the computing continuum: Fluid function execution in heterogeneous computing environments." In *Heterogeneity in Computing Workshop at IPDPS* (IPDPSW) pp. 66-75. 2021.

DELTA+: Costs and Constraints

- Distribute tasks to different resources with different costs
- Relating cost vs. time
  - Different resources have different optimization criteria
- User definition
  - Allows for expression of tradeoffs and weighting
    - E.g., $1 ≈ 10 minutes
  - Costs at different levels
- Constraint satisfaction
  - Cost budgets
  - Time budgets
  - HPC Allocations
- How do we optimize under constraints?

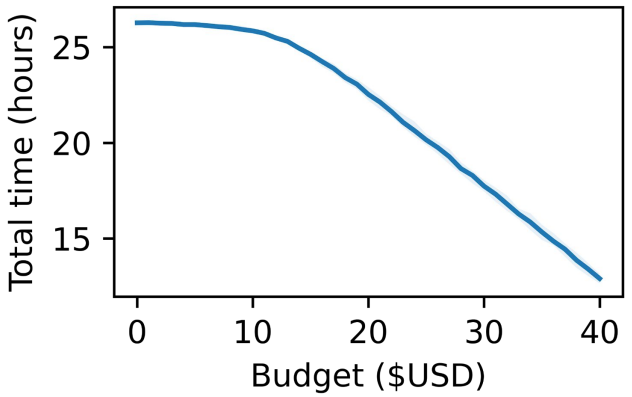
DELTA+: Optimization

- Deadlines
  - Allows for prioritization of faster or more time-effective resources for task allocation
  - When do you need the compute done?
- Cost constraints
  - Absolute budgets
  - Prioritizes cost-effective resources
- Automated optimization definition
  - Multiple constraints make resource selection difficult
  - Solution: select resources to maximize the likelihood of workload completion under constraint



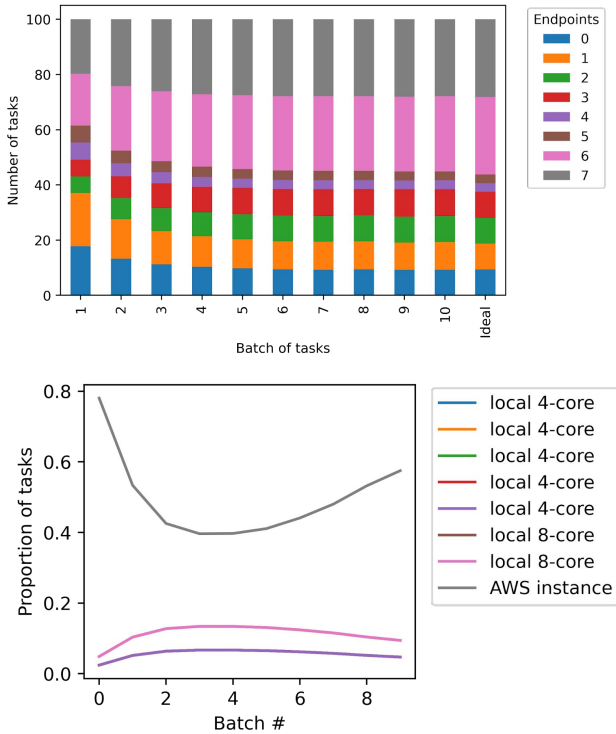
DELTA+: Resource Provisioning using funcX

- AWS Resource Provider
- Plug and play with DELTA+
  - Expose the provider by associating it with an endpoint and assigning relevant costs
- Nonlinear tradeoffs due to overhead, imperfect scaling, and time costs



DELTA+: Experimental Results

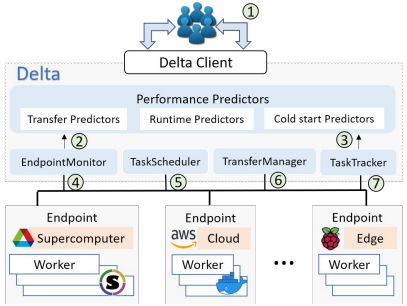
- Incorporate probabilistic task placement
- Experimental setup
  - Simulated 8 endpoints of various costs and performance levels
  - Costs initially known, performance hidden
  - Launched 10 batches of 100 tasks
  - Optimized for performance per dollar
- Results
  - **Learned relative performance to within 10% of optimal in 5 batches**



Current and Future Development

- Incorporate more diverse resource providers for HPC and cloud resources
- Extend Delta+ further into the edge
  - Port compute request functionality to extreme LP devices
  - Managed function execution across a hierarchical ecosystem
- Develop heuristic free control framework that exceeds probabilistic allocation
- Large scale deployments
  - Using resource providers, test Delta+'s capabilities at 1,000+ endpoint scale
- Automate installation and configuration
  - Manage resource and worker configuration using funcX

Conclusion



- Serverless solves many problems but introduces new complexities to overcome
- Task placement is limited by the human element
- We can automate task distribution and remove the need for humans in optimization metrics and resource selection
- Delta+ relies on funcX to serve as a "serverless anywhere" framework to enable the use of modern distribution and heterogeneity

Contact—mbaughman@uchicago.edu