

# A Cache-based Approach to Dynamic Switching between Different Dataflows in Crowdsourcing

Yusuke Suzuki<sup>†</sup>, Masaki Matsubara<sup>†</sup>, Keishi Tajima<sup>‡</sup>, Toshiyuki Amagasa<sup>†</sup>, Atsuyuki Morishima<sup>†</sup>

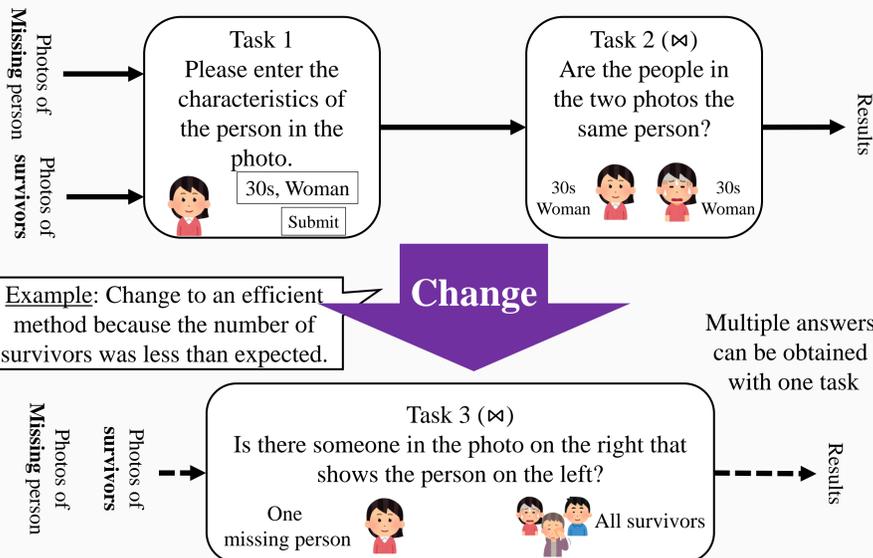


## Background

We may want to change dataflows halfway.

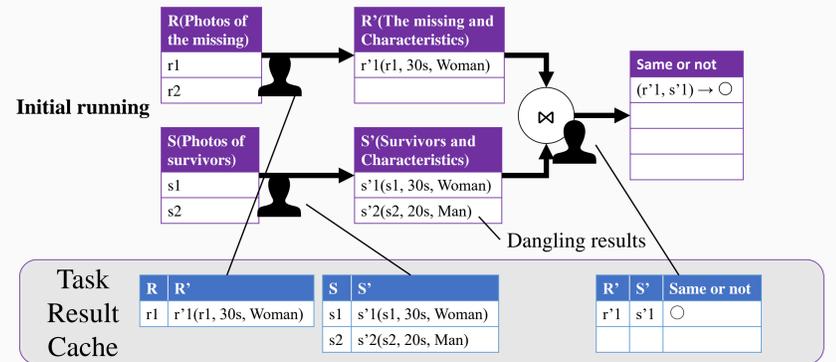
Dataflow Example:

Identifying the missing and survivors when a disaster occurs

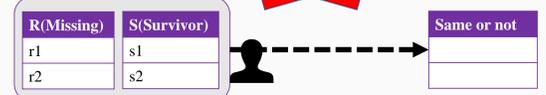


## Problem & Purpose

Existing method cannot cope with the changing dataflows[1].



Changing dataflows & Running



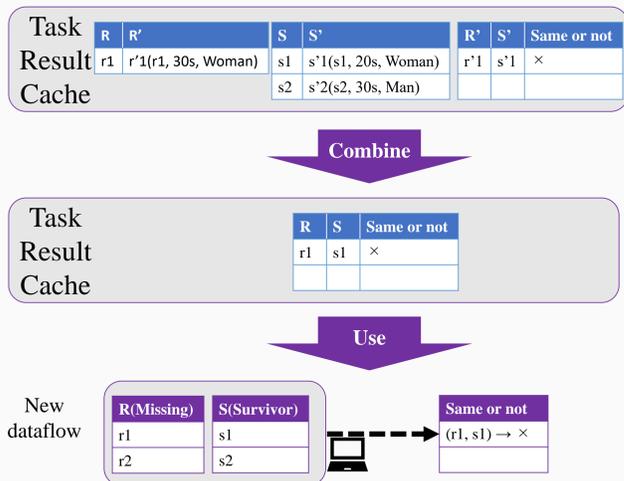
Purpose

To reduce the total monetary cost in the dataflow change process

[1] G. Little, L. B. Chilton, M. Goldman, and R. C. Miller, "Turkit: human computation algorithms on mechanical turk," in Proc. of ACM UIST 2010, 2010, pp. 57–66.

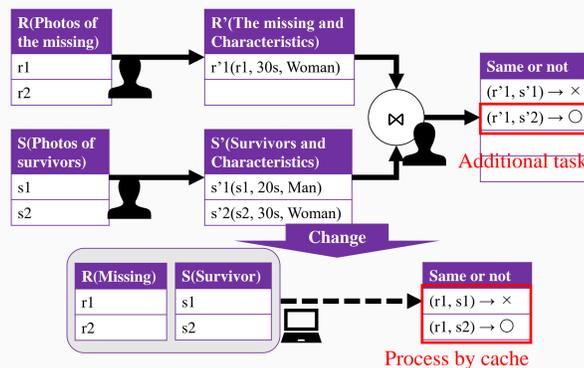
## Proposed Method Propose a rerunning method in changing dataflows.

### Combining Caches



Issue additional tasks to use dangling results.

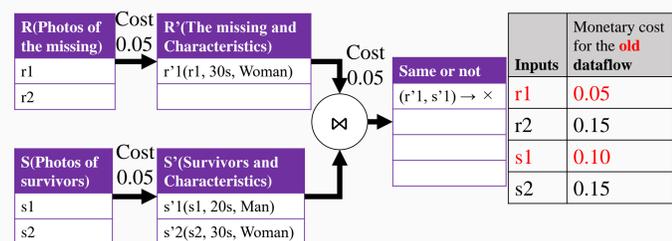
However, there is a tradeoff between cost reduction by cache and the cost increase by additional tasks.



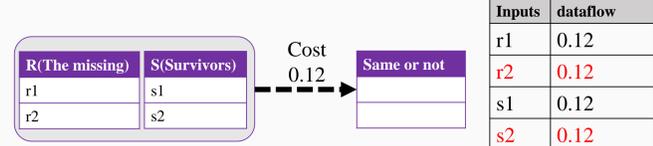
### Cost Estimation

Compare the costs of the two plans

◆ Monetary cost for the old dataflow



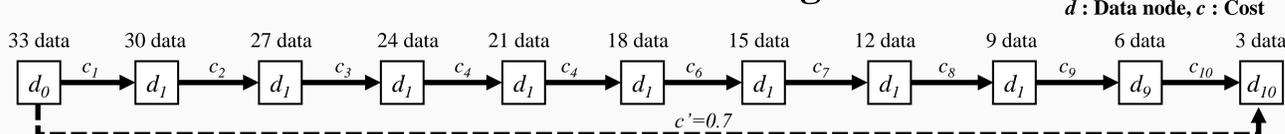
◆ Monetary cost for the new dataflow



## Simulation

It is possible to identify the best point to minimize the total cost and there is no obvious solution.

### Simulation settings



Num of tasks

10

Data location at changing

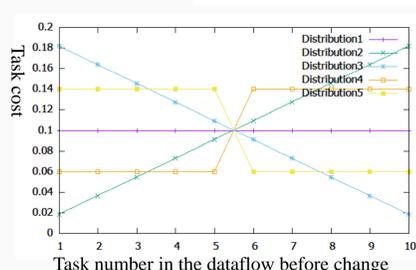
3 dangling results for each data node

Processing order

Process from data closest to the last node

Task cost distribution

Total cost of the task in the dataflow before change : 1.0  
The task cost in the dataflow after change : 0.7



Cost distribution

- 1 : Constant
- 2 : Linear increase
- 3 : Linear decrease
- 4 : Low & High
- 5 : High & Low

### Simulation Results

