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

Yusuke Suzuki†, Masaki Matsubara†, Keishi Tajima‡, Toshiyuki Amagasa†, and Atsuyuki Morishima†
Key points of the presentation

1. 【Background】 Crowdsourcing sometimes makes dataflow change halfway, but it costs a lot of money.
2. 【Related Work】 With the method using task result caches, it cannot cope with dataflow change.
3. 【Our Approach】 We propose the method to use caches coping with changing dataflows.
4. 【Simulation】 Our simulation results showed that it is possible to identify the best point to minimize the total cost.
Background (1/4): Crowdsourcing Takes Time and Monetary Cost

Crowdsourcing Characteristics

Because of slow processing by human, it requires time and monetary costs.

Task 1
Please translate the following sentence from Japanese to English.

“Konnichiwa”

Worker

Task 2
Please translate the following sentence from English to Spanish.

“Hello”

Worker

Japanese

$0.10 per task

$0.05 per task

Spanish

Japanese

English

Spanish
Because completing all tasks takes long time, we may want to change the dataflow halfway due to various factors.

Task 1: Please translate the following sentence from Japanese to English

“Konnichiwa”

Example: Japanese language data to be translated increased, and huge money cost was expected to incur.

Task 2: Please translate the following sentence from English to Spanish

“Hello”

Task 3: Please translate the following sentence from Japanese to Spanish

“Konnichiwa”
Background (3/4):
Changing dataflows in crowdsourcing are difficult

Switching query plans at appropriate timing. (Eddies[1] et al.)

→ Since it is necessary to develop a mechanism for each type of operator, such a mechanism cannot be generalized for crowdsourcing settings where we have variety of tasks

Propose a rerunning method with the task result cache.
→ It is reasonable because the cost of the computer involved in using caches is low compared to the cost of processing tasks in crowdsourcing.

The purpose of our research:
To reduce the total monetary cost in the dataflow change process.

Background(4/4):
It takes extra money cost to redo the task.
Changing the dataflow makes it impossible to use the result of the task performed halfway and the cost for that is wasted.

The diagram illustrates the sequence of tasks and the consequences of changing the dataflow. Tasks 1 and 2 are in English, and Task 3 is in Japanese. The change is shown to affect the availability of the task results and the monetary cost.

The purpose of our research:
To reduce the total monetary cost in the dataflow change process.
Key points of the presentation

1. 【Background】 Crowdsourcing sometimes makes dataflow change halfway, but it costs a lot of money.
2. 【Related Work】 With the method using task result caches, it cannot cope with dataflow change.
3. 【Our approach】 We propose the method to use caches coping with changing dataflows.
4. 【Simulation】 Our simulation results showed that it is possible to identify the best point to minimize the total cost.
Existing Method: Rerunning method with the task result caches [2]

It can resume halfway when crashing

<table>
<thead>
<tr>
<th>Japanese</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>Konnichiwa</td>
<td>Hello</td>
</tr>
<tr>
<td>Arigatou</td>
<td>Thanks</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>English</th>
<th>Spanish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hello</td>
<td>Hola</td>
</tr>
</tbody>
</table>

Initial running

Process by human

Crash

Create

Cache

Rerunning

Process by cache

Process by cache

Existing Method: Cannot use caches after changing the dataflow[2]

Because caches do not correspond to the task in new dataflow, the caches cannot be used.

Key points of the presentation

1. 【Background】 Crowdsourcing sometimes makes dataflow change halfway, but it costs a lot of money.

2. 【Related Work】 With the method using task result caches, it cannot cope with dataflow change.

3. 【Our Approach】 We propose the method to use caches coping with changing dataflows.

4. 【Simulation】 Our simulation results showed that it is possible to identify the best point to minimize the total cost.
Combining caches

Combine cache from task input and task output

Initial running

<table>
<thead>
<tr>
<th>Cache</th>
<th>Japanese</th>
<th>English</th>
<th>Spanish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Konnichiwa</td>
<td>Hello</td>
<td></td>
<td>Hola</td>
</tr>
<tr>
<td>Arigatou</td>
<td>Thanks</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Combine

<table>
<thead>
<tr>
<th>Cache</th>
<th>Japanese</th>
<th>Spanish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Konnichiwa</td>
<td>Hola</td>
<td></td>
</tr>
</tbody>
</table>

Changing a dataflow + Rerunning

<table>
<thead>
<tr>
<th>Japanese</th>
<th>Spanish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Konnichiwa</td>
<td>Hola</td>
</tr>
<tr>
<td>Arigatou</td>
<td></td>
</tr>
</tbody>
</table>
Processing data without corresponding cache

In the old dataflow, only data with no corresponding cache will be processed.

New dataflow

- Process by cache. Does not require cost.
- Process by human. Requires more cost.

Old dataflow

- Japanese
  - konnichiwa
  - Arigatou

- English
  - Hello
  - Thanks

- Spanish
  - Hola

Dangling result

Create

Cache

- Japanese
  - konnichiha
- Spanish
  - Hola

Use
Process the dangling result in the old dataflow

Reduce wasted task results by executing additional tasks on dangling results.

Old dataflow

Japanese
Konnichiwa
Arigatou

Dangling result

English
Hello
Thanks

Spanish
Hola

Additional task

Cache

<table>
<thead>
<tr>
<th>Japanese</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>Konnichiwa</td>
<td>Hello</td>
</tr>
<tr>
<td>Arigatou</td>
<td>Thanks</td>
</tr>
</tbody>
</table>

Combine

Result of additional task

Cache

<table>
<thead>
<tr>
<th>Japanese</th>
<th>Spanish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Konnichiwa</td>
<td>Hola</td>
</tr>
<tr>
<td>Arigatou</td>
<td>Gracias</td>
</tr>
</tbody>
</table>
Relationship between dangling results and costs

It is not enough to simply process dangling results to the end.

Compare the costs and process with the cheaper dataflow.

Plan 1
Process the data in the **OLD** dataflow

Plan 2
Process the data in the **NEW** dataflow

**Dangling result**

**Comparing costs:**
- Additional processing cost in dataflow before change
- Processing cost in dataflow after change

**Already processed**
Cost estimation for each dataflow

### Monetary cost for the old dataflow

<table>
<thead>
<tr>
<th>Input</th>
<th>Monetary cost for the old dataflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Konnichiwa</td>
<td>0.00</td>
</tr>
<tr>
<td>Arigatou</td>
<td>0.05</td>
</tr>
<tr>
<td>Sayounara</td>
<td>0.15</td>
</tr>
</tbody>
</table>

### Monetary cost for the new dataflow

<table>
<thead>
<tr>
<th>Input</th>
<th>Monetary cost for the new dataflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Konnichiwa</td>
<td>0.10</td>
</tr>
<tr>
<td>Arigatou</td>
<td>0.10</td>
</tr>
<tr>
<td>Sayounara</td>
<td>0.10</td>
</tr>
</tbody>
</table>
Key points of the presentation

1. 【Background】 Crowdsourcing sometimes makes dataflow change halfway, but it costs a lot of money.

2. 【Related Work】 With the method using task result caches, it cannot cope with dataflow change.

3. 【Our Approach】 We propose the method to use caches coping with changing dataflows.

4. 【Simulation】 Our simulation results showed that it is possible to identify the best point to minimize the total cost.
Simulation purpose and Results

Purpose: To show relationship between data processing cost and cost before change

Results:
• it is possible to identify the best point to minimize the total cost.
• There are no obvious solutions.

The number of the data processed in the old dataflow
Summary

• We proposed the method to use caches corresponding to changing dataflows.
• Our simulation showed that it is possible to identify the best point to minimize the total cost.
• I explored the dataflow including join operation. (poster)

Future Work

• Since cost estimation is costly for calculation,
• we will consider a method to reduce calculation cost.