Data input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Row id | event | venue | place | currency | Currency\_symbol |
| 0 | BREAKFAST | COMMERCIAL | RMS CAMPANIA | Shilings | s |
| 1 | LUNCHEON | SOC; | BOULEVARD, 66TH AND 67TH STREETS | Dollars | $ |
| 2 | DINNER(?) | COMM. | (60 PINE ST., NEW YORK, NY) | Dollars |  |
| 3 | dinner | [SOC?]; | NEW YORK | Canadian Dollars |  |
| 4 | lunch | COM | PHILADELPHIA, PA. | Deutsche Marks |  |
| 5 | Breakfast | COMMERCIAL | TAMPA, FL | Dollars | $ |

**Merge Example I: Auto-Merging Disjoint Changes**

Process I (P1):

event 🡪 Upper() 🡪 event1 🡪 remove\_char() 🡪 event2

Delta Changes by Operations in P1:

|  |  |  |
| --- | --- | --- |
| Step ID | Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Uppercase | (3, ‘event’)  (4, ‘event’)  (5, ‘event’) |
| 2 | Remove characters | (2, ‘event’) |

Process II (P2):

venue 🡪 remove\_char() 🡪 venue1. 🡪 value\_repl. 🡪 venue2

|  |  |  |
| --- | --- | --- |
| Step ID | Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Remove characters | (1, ‘venue’)  (2, ‘venue’)  (3, ‘venue’) |
| 2 | Value replacement | (2, ‘venue’)  (4, ‘venue’) |

Merge P1 (N steps) and P2 (M steps)

Conflict-free I

|  |  |  |
| --- | --- | --- |
| Step ID | P1.Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Uppercase | (3, ‘event’)  Conflict-free I  (4, ‘event’)  Conflict-free I  (5, ‘event’) |
| 2 | Remove characters | (2, ‘event’)  Conflict-free I |

|  |  |  |
| --- | --- | --- |
| Step ID | P2.Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Remove characters | (1, ‘venue’)  (2, ‘venue’)  (3, ‘venue’) |
| 2 | Value replacement | (2, ‘venue’)  (4, ‘venue’) |

Process of conflict detecting:

1. Enumerate operations in P1, compare each with the operation in P2
2. Conflict could occur during the comparison in-between
3. We need to detect: N\*M times.

If [[], [], [], [],…] is returned, which means current P1 and P2 can be auto-merged without conflicts (the changes are disjoint).

**Merge Example II: Auto-Merging Commutative Changes**

Process I (P1):

event 🡪 Upper() 🡪 event1 🡪 remove\_char() 🡪 event2

Delta Changes by Operations in P1:

|  |  |  |
| --- | --- | --- |
| Step ID | Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Uppercase | (3, ‘event’)  (4, ‘event’)  (5, ‘event’) |
| 2 | Remove characters | (2, ‘event’) |

Process II (P2):

event🡪strip\_whitesp()🡪 event3 🡪value\_repl() 🡪 event4

Delta Changes by Operations in P2:

|  |  |  |
| --- | --- | --- |
| Step ID | Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Strip whitespaces | (3, ‘event’)  (5, ‘event’) |
| 2 | Value replacement | (3, ‘event’)  (5, ‘event’) |

Merge P1 (N steps) and P2 (M steps)

Conflict -freeI

|  |  |  |
| --- | --- | --- |
| Step ID | P1.Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Uppercase | (3, ‘event’)  Conflict? I  (4, ‘event’)  Conflict? I  (5, ‘event’) |
| 2 | Remove characters | (2, ‘event’)  Conflict-free I |

|  |  |  |
| --- | --- | --- |
| Step ID | P2.Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Strip whitespaces | (3, ‘event’)  (5, ‘event’) |
| 2 | Value replacement | (3, ‘event’)  (5, ‘event’) |

We mark *Conflict?* For the relationship between Uppercase() in P1, and Strip\_whitesp() in P2.

Rationale: various operations (from separate processes) work on the same region which result in different repairing results.

Solution:

* Re-apply Uppercase() and Strip\_whitesp() in two orders, and find that the data results are the same.
  + This means Uppercas() and Strip\_whitesp() can be merged in a commutative way
* Merge(P1.op1, P2.op1) 🡺 execute updated data changes
* Question: No need to compare P1.op1 and P2.op2 ?! Because P2.op2 must be executed after P2.op1.

Merge Example III: Priority-Based Merging

Merge Example IV: User-Defined Conflict Resolution

|  |  |  |
| --- | --- | --- |
| Row id | currency | currency\_symbol |
| 0 | Shilings | s |
| 1 | Dollars | $ |
| 2 | Dollars |  |
| 3 | Canadian Dollars |  |
| 4 | Deutsche Marks |  |
| 5 | Dollars |  |

Process I (P1):

(currency, currency\_symbol ) 🡪 value\_repl() 🡪 currency\_symbol1

Delta Changes by Operations in P1:

|  |  |  |
| --- | --- | --- |
| Step ID | Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Value replacement | (2, ‘currency\_symbol’)  (3, ‘currency\_symbol’)  (4, ‘currency\_symbol’)  (5, ‘currency\_symbol’) |

Process II (P2):

currency\_symbol🡪remove\_col()🡪 NULL

Delta Changes by Operations in P2:

|  |  |  |
| --- | --- | --- |
| Step ID | Operation | diff(eventi, eventi+1):  (row id, column name) |
| 1 | Remove column | (0, ‘currency\_symbol’)  (1, ‘currency\_symbol’)  (2, ‘currency\_symbol’)  (3, ‘currency\_symbol’)  (4, ‘currency\_symbol’)  (5, ‘currency\_symbol’) |