Parallel Linkage

Hung-sik Kim[1] hungsik@psu.edu
Dongwon Lee[1,2] dongwon@psu.edu

[1]Computer Science and Engineering
[2]Information Sciences and Technology
The Pennsylvania State University
University Park, PA, 16802, USA

Record Linkage

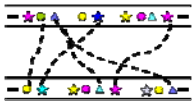
- **Record Linkage**: Identify all matching records between two collections of records
  - Citations in digital libraries
  - Addresses in customer relationship management
  - Patient records in health information system
- Different from Database Join
  - Whole record vs. a few short attributes
  - Match vs. Match & Merge

Technical Landscape

- Abundant research in many disciplines
- Also relevant to:
  - DB: approximate join, merge/purge
  - DL: citation matching, de-duplication, author name disambiguation
  - AI: identity matching
  - NLP: word sense disambiguation
  - IR: web query results clustering
  - LIS: name authority control

Related Work

- SERF project @ Stanford Univ.
  - Blackbox approach
  - Distributed and parallel record linkage
- Febrl project @ Australia National Univ.
  - Open source GUI based record linkage tool
- Data Linkage project @ Penn State Univ.
  - New applications
    - Group Linkage, Video Linkage
  - New aspects
    - Adaptability
  - http://pike.psu.edu/linkage/

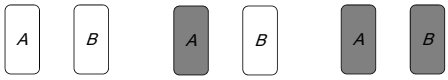


Data Set

- Set characteristic
  - Clean set
    - *A is clean iff* $\forall r,s \in A,\ r \not\approx s$
  - Dirty set
    - *A is dirty iff* $\exists r,s \in A,\ r \approx s$
- Input data sets

O(|A||B|)

Clean vs. Clean | Dirty vs. Clean | Dirty vs. Dirty

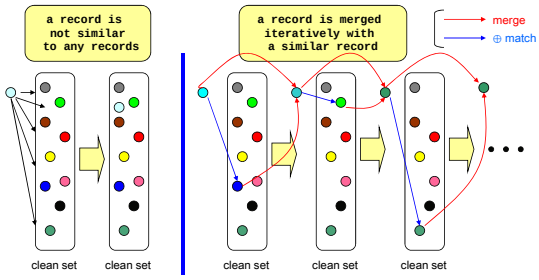


Type of Matching & Merging

- match(r,s) returns
  - $r \approx s$
    - $r \sqsubseteq s$  → merge(*r*,*s*) = *s*
    - $r \sqsupseteq s$  → merge(*r*,*s*) = *r*
    - $r \equiv s$  → merge(*r*,*s*) = *r*
    - $r \oplus s$  → merge(*r*,*s*) = $r \cup s$
  - $r \not\approx s$ → no merge

Iterative Linkage Framework

- *match()*, *merge()* and iteration

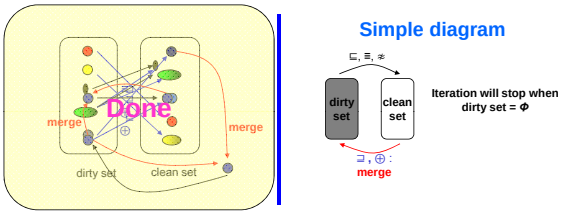


Proposed Algorithms

- Sequential Linkage: six variants
  - *Clean-Clean: S-CC*
  - *Dirty-Clean: S-DC, s-DC$_{self}$*
  - *Dirty-Dirty: s-DD1, s-DD2, s-DD3*
- Parallel Linkage: five variants
  - *Clean-Clean: p-CC*
  - *Dirty-Clean: p-DC, p-DC$_{self}$*
  - *Dirty-Dirty: p-DD1, p-DD2*

Sequential Linkage

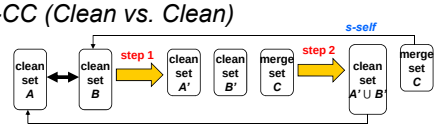
- s-DC (Dirty vs. Clean)



- *s-self* (one dirty set)
  - same as *s-DC*, but start with empty clean set

Sequential Linkage

- *s-CC (Clean vs. Clean)*

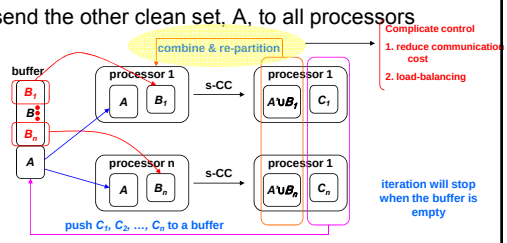


  - iteration will stop when $C = \Phi$

- *s-DC$_{self}$(A,B) = s-CC(s-self(A),B)*
- *s-DD1(A,B) =s-DC(A,s-self(B))*
- *s-DD2(A,B) =s-CC(s-self(A),s-self(B))*
- *s-DD3(A,B) =s-self(A∪B)*

Parallel Linkage

- *p-CC*
  - put all clean sets into a buffer
  - partition one clean set, B, and send subsets
  - send the other clean set, A, to all processors



Parallel Linkage

- *p-DC:* partition data sets and apply *s-DC*
- *p-DC$_{self}$:* partition data sets and apply *s-DC$_{self}$*
- *p-DD1:* partition data sets and apply *s-DD1*
- *p-DD2:* partition data sets and apply *s-DD2*

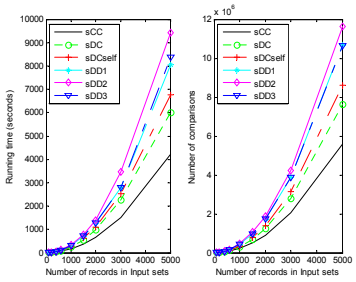
- *Similar implementation to p-CC (but much more complicated)*
  - Details omitted in the presentation

Experimental Setup

- Evaluation metrics
  - sequential linkage
    - number of comparisons (NC)
    - running time (RT)
  - parallel linkage
    - $speedup_{NC}$ / $efficiency_{NC}$
    - $speedup_{RT}$ / $efficiency_{RT}$
  - Data set
    - size : up to 5,000 records
  - Environment
    - distributed Matlab in PC cluster at Penn State
    - 1~32 nodes

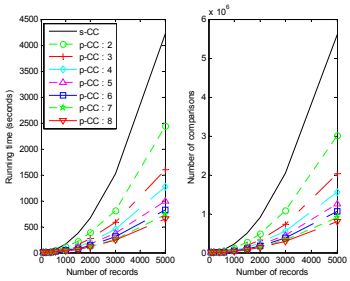
Sequential Linkage

- Running time and Number of comparisons



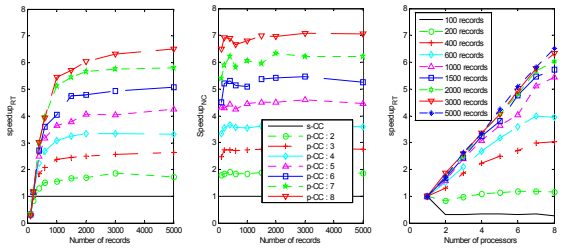
p-CC

- Running time and Number of comparisons



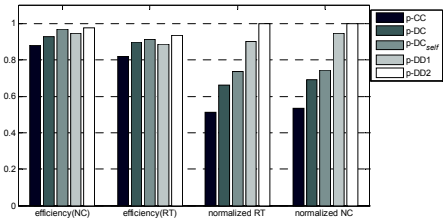
p-CC

- Speedup and Scalability



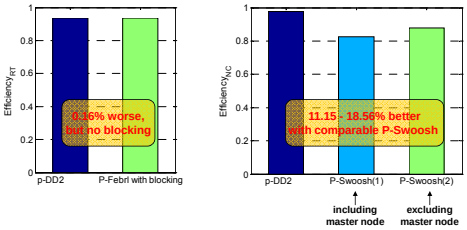
Parallel Linkage

- Efficiency and Normalized RT/NC



Comparison against Others

- $Efficiency_{RT}$ and $Efficiency_{NC}$



Conclusion

- Conclusion
  - Sequential linkage methods can be selected according to input data characteristics
  - Parallel linkage methods are shown to be efficient and scalable
  - Any distance function can be used in sequential or parallel linkage methods

Any Questions?

- Data/Codes are downloadable at:
  - **http://pike.psu.edu/download/cikm07/**

Thank You !