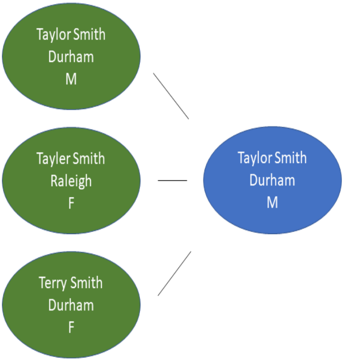
Efficient and Scalable Bipartite Matching through Fast Beta Linkage (fabl)

Brian Kundinger

Introduction to Record Linkage

Fellegi and Sunter

A B Γ



$n_A \times F$ $n_B \times F$

$$\gamma_{11} = \{1, 1, 1, 1\}$$

$$\gamma_{21} = \{2, 1, 2, 2\}$$

$$\gamma_{31} = \{3, 1, 1, 2\}$$

$n_A n_B \times F$

Notation

- File A with records indexed $i \in \{1, \ldots, n_A\}$ and file B with records $j \in \{1, \ldots, n_B\}$. We use F features for linkage, with L_f possible levels of agreement on feature f.
- $\Gamma \in \mathbb{R}^{n_A n_B \times F}$ matrix of comparison vectors where $\gamma_{ij}^f \in \{1, \ldots, L_f\}$
- $Z_j = \begin{cases} i, & \text{if records } i \in A \text{ and } j \in B \text{ refer to the same entity;} \\ n_A + 1, & \text{if record } j \in B \text{ does not have a match in file } A; \end{cases}$
- $m_{fl} = P\left(\gamma_{ij}^f = l | Z_j = i\right)$
- $u_{fl} = P\left(\gamma_{ij}^f = l | Z_j \neq i\right)$
- $\lambda = P(Z_j \leq n_A)$

m's and u's

Previous work

- Fellegi and Sunter (1969)
  - independent classification of all $n_A n_B$ record pairs. Transitive closure acheived through postprocessing
- Enamorado et al (2019): `fastlink`
  - provides efficient and scalable FS model
- Sadinle (2017) - Beta Record Linkage (`BRL`)
  - Bayesian method for bipartite matching, strictly enforces one-to-one matching. Does not scale well to large linkage tasks

Fast Beta Linkage (fabl)

$$P(\Gamma|\mathbf{Z}, \mathbf{m}, \mathbf{u}) = \prod_{j=1}^{n_B} \prod_{i=1}^{n_A} \left[\prod_{f=1}^{F} \prod_{l=1}^{L_f} m_{fl}^{I(Z_j=i)} u_{fl}^{I(Z_j \neq i)} \right]^{I(\gamma_{ij}^f = l)}$$

$$\mathbf{m_f} \sim \text{Dirichlet}(\alpha_{f1}, \ldots, \alpha_{fL_f})$$

$$\mathbf{u_f} \sim \text{Dirichlet}(\beta_{f1}, \ldots, \beta_{fL_f})$$

$$Z_j|\lambda = \begin{cases} \frac{1}{n_A}\lambda & z_j \leq n_A; \\ 1 - \lambda & z_j = n_A + 1 \end{cases}$$

$$\lambda \sim \text{Beta}(\alpha_\lambda, \beta_\lambda)$$

- Model specification allows for **parallel/distributed** computing, **hashing** of comparison vectors, and **storage efficient indexing (SEI)**

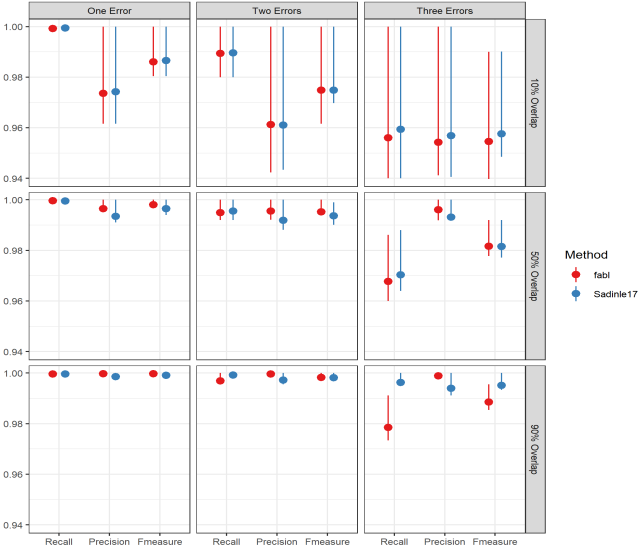
Hashing

- Recognize there are at most $P = \prod_{f=1}^{F} L_f$ unique agreement patterns. When (i, j) pair exhibits agreement pattern p, say $(i, j) \in h_p$
- Reduce data to sufficient statistics
  - $H_p = \sum_{i,j} \mathbf{1}_{(i,j) \in h_p}$
  - $H_{p_j} = \sum_{i} \mathbf{1}_{(i,j) \in h_p}$
- Gibbs sampler over each record pair has complexity $O(n_A \times n_B \times F)$, but sampler over agreement pattern has complexity $O(P \times n_B \times F)$
- Speeds up posterior calculations and sampling

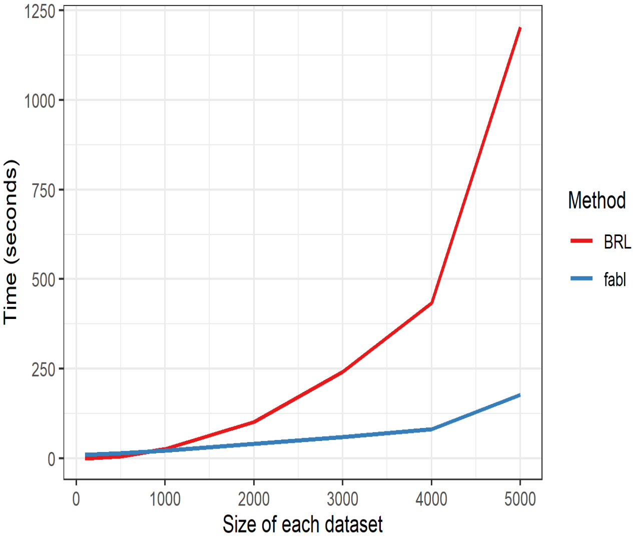
Managing Large Data

- **Distributed Computing** - Partition data in to chunks $\{A_I\}$ and $\{B_J\}$. Compare records, hash results, compute summary statistics in parallel, and synthesize results
- **Storage Efficient Indexing (SEI)** - Remove highly unlikely record labels from memory. Proper weights for calculations maintained through summary statistics $\{H_p\}$ and $\{H_{p_j}\}$

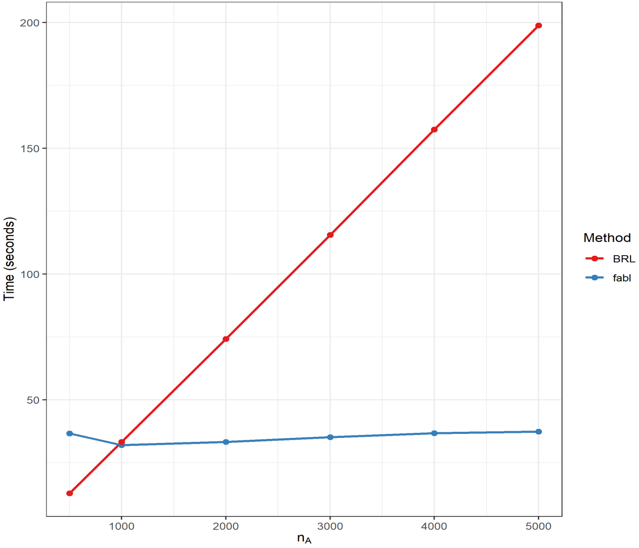
Simulation Results 1



Simulation Results 2



Simulation Results 3



Extentions and Directions

- Linkage in the face of duplicates within datasets
- Models that allow reliability of information to differ by subgroup in the data
- Linkage over blocked data (allows for much larger linkage tasks)

Questions?

Additional Slides

El Salvador

Distributed computation

Storage Efficient Indexing