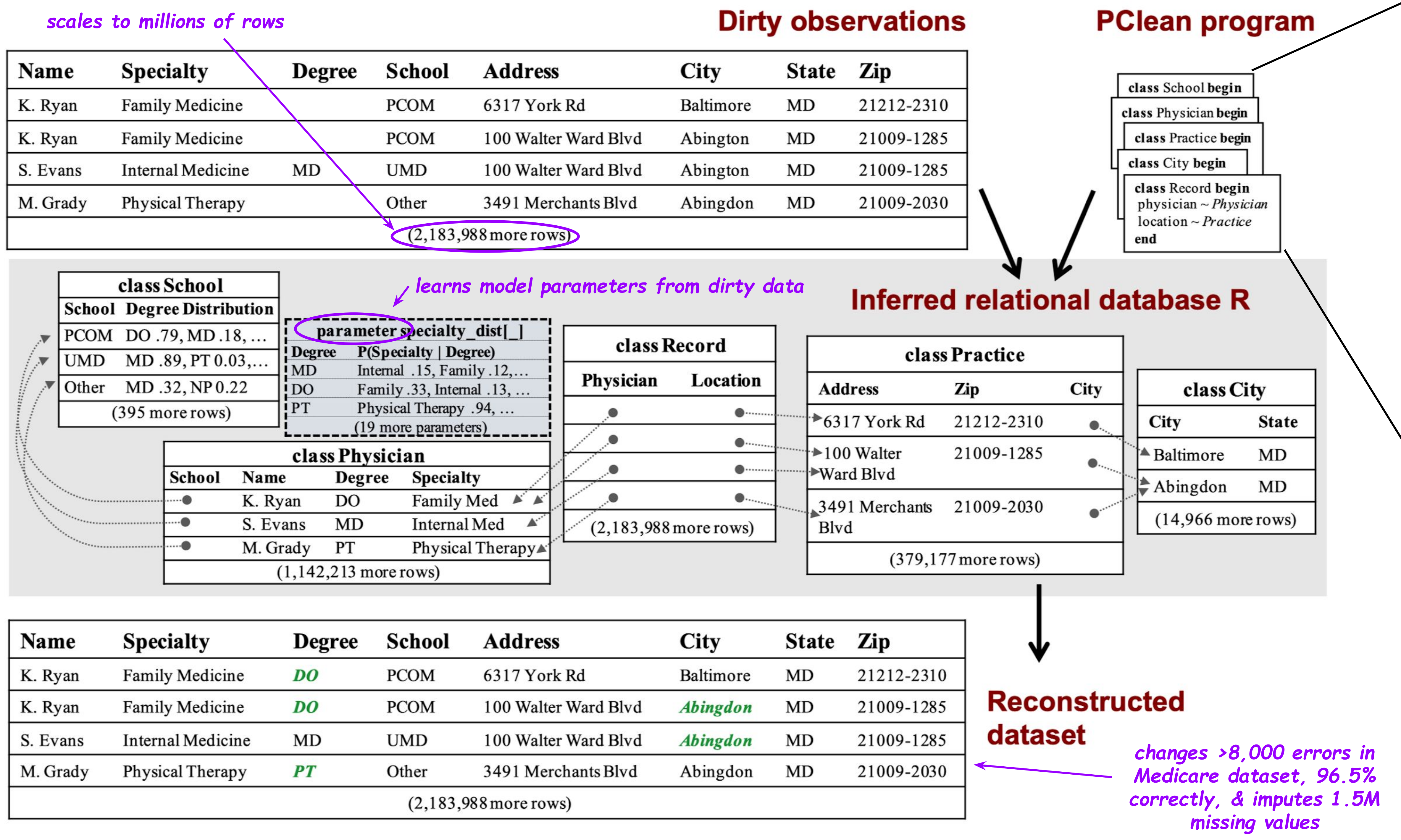
PClean: Bayesian Data Cleaning at Scale via Domain-Specific Probabilistic Programming

Alexander K. Lew, Monica Agrawal, David Sontag, Vikash K. Mansinghka
MIT

1. Overview

PClean is a domain-specific probabilistic programming language for inferring ground-truth relational databases from flat, dirty datasets.



scales to millions of rows

Dirty observations

Name	Specialty	Degree	School	Address	City	State	Zip
K. Ryan	Family Medicine		PCOM	6317 York Rd	Baltimore	MD	21212-2310
K. Ryan	Family Medicine		PCOM	100 Walter Ward Blvd	Abington	MD	21009-1285
S. Evans	Internal Medicine	MD	UMD	100 Walter Ward Blvd	Abington	MD	21009-1285
M. Grady	Physical Therapy		Other	3491 Merchants Blvd	Abingdon	MD	21009-2030
			(2,183,988 more rows)				

PClean program

```
class School begin
class Physician begin
class Practice begin
class City begin
class Record begin
 physician ~ Physician
 location ~ Practice
end
```

Inferred relational database R

learns model parameters from dirty data

class School

School	Degree Distribution
PCOM	DO .79, MD .18, …
UMD	MD .89, PT 0.03,…
Other	MD .32, NP 0.22
(395 more rows)	

parameter specialty_dist[_]

Degree	P(Specialty | Degree)
MD	Internal .15, Family .12,…
DO	Family .33, Internal .13, …
PT	Physical Therapy .94, …
(19 more parameters)	

class Physician

School	Name	Degree	Specialty
●	K. Ryan	DO	Family Med
●	S. Evans	MD	Internal Med
●	M. Grady	PT	Physical Therapy
(1,142,213 more rows)			

class Record

Physician	Location
●	●
●	●
●	●
●	●
(2,183,988 more rows)	

class Practice

Address	Zip	City
6317 York Rd	21212-2310	●
100 Walter Ward Blvd	21009-1285	●
3491 Merchants Blvd	21009-2030	●
(379,177 more rows)		

class City

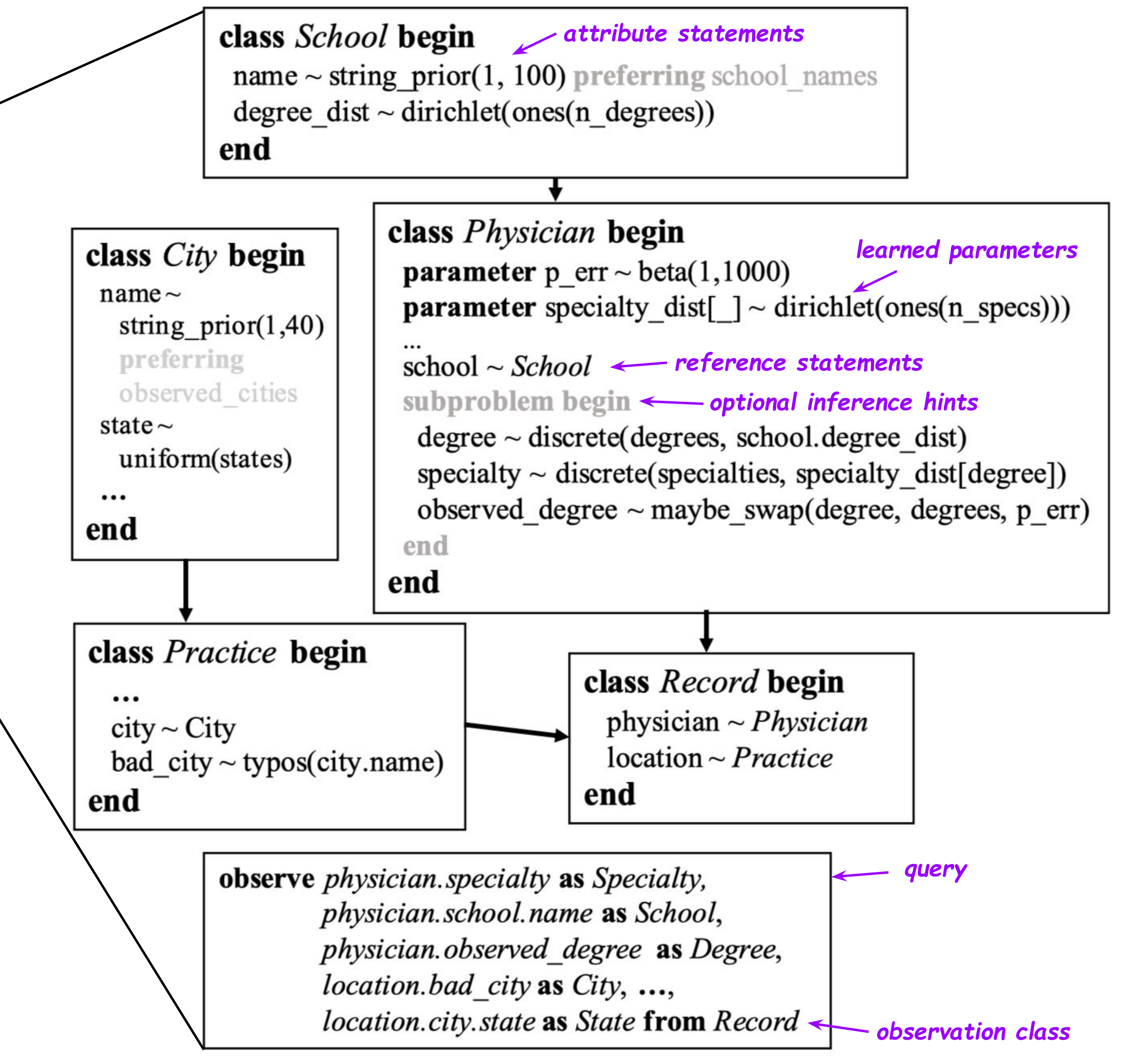
City	State
Baltimore	MD
Abingdon	MD
(14,966 more rows)	

Reconstructed dataset

Name	Specialty	Degree	School	Address	City	State	Zip
K. Ryan	Family Medicine	***DO***	PCOM	6317 York Rd	Baltimore	MD	21212-2310
K. Ryan	Family Medicine	***DO***	PCOM	100 Walter Ward Blvd	***Abingdon***	MD	21009-1285
S. Evans	Internal Medicine	MD	UMD	100 Walter Ward Blvd	***Abingdon***	MD	21009-1285
M. Grady	Physical Therapy	***PT***	Other	3491 Merchants Blvd	Abingdon	MD	21009-2030
			(2,183,988 more rows)				

changes >8,000 errors in Medicare dataset, 96.5% correctly, & imputes 1.5M missing values

2. PClean Modeling Language



```
class School begin
  name ~ string_prior(1, 100) preferring school_names
  degree_dist ~ dirichlet(ones(n_degrees))
end
```

attribute statements

```
class City begin
  name ~
    string_prior(1,40)
    preferring
    observed_cities
  state ~
    uniform(states)
  ...
end
```

```
class Physician begin
  parameter p_err ~ beta(1,1000)
  parameter specialty_dist[_] ~ dirichlet(ones(n_specs)))
  ...
  school ~ School
  subproblem begin
    degree ~ discrete(degrees, school.degree_dist)
    specialty ~ discrete(specialties, specialty_dist[degree])
    observed_degree ~ maybe_swap(degree, degrees, p_err)
  end
end
```

learned parameters

reference statements

optional inference hints

```
class Practice begin
  ...
  city ~ City
  bad_city ~ typos(city.name)
end
```

```
class Record begin
  physician ~ Physician
  location ~ Practice
end
```

```
observe physician.specialty as Specialty,
        physician.school.name as School,
        physician.observed_degree as Degree,
        location.bad_city as City, ...,
        location.city.state as State from Record
```

query

observation class

Equivalent view: in topological order, simulate infinite collection of objects from each class from Pitman-Yor

Left: A PClean program defines a **relational schema for a database of objects underlying the dirty data**, along with a **probabilistic relational model** over object attributes.

Below: PClean uses a **domain-general non-parametric structure prior** over the number of objects of each class, and over their relationships.

```
GenerateSkeleton($\mathcal{C}, |\mathbf{D}|$):
    ▷ Create one $C_{obs}$ object per observed record
    $\mathbf{S}_{C_{obs}} := \{1, \dots, |\mathbf{D}|\}$
    ▷ Generate a class after all referring classes:
    for class $C \in \textsc{TopoSort}(\mathcal{C} \setminus \{C_{obs}\})$ do
        ▷ Collect references to class $C$
        $\mathbf{Refs_S}(C) := \{(r, Y) \mid r \in \mathbf{S}_{C'}, T(C'.Y) = C\}$
        ▷ Generate targets of those references
        $\mathbf{S}_C \sim \textsc{GenerateObjectSet}(C, \mathbf{Refs_S}(C))$
        ▷ Assign reference slots pointing to $C$
        for object $r' \in \mathbf{S}_C$ do
            for referring object $(r, Y) \in r'$ do
                $r.Y := r'$
    ▷ Return the skeleton
    return $\{\mathbf{S}_C\}_{C \in \mathcal{C}}, (r, Y) \mapsto r.Y$
```

```
GenerateObjectSet($C, \mathbf{Refs_S}(C)$):
    $s_C \sim Gamma(1,1)$; $d_C \sim Beta(1,1)$
    ▷ Partition $\mathbf{Refs_S}(C)$ into disjoint co-referring
      subsets; each represents an object
    $\mathbf{S}_C \sim CRP(\mathbf{Refs_S}(C), s_C, d_C)$
```

3. PClean Inference Engine

1) Initializes latent database with *per-observation* SMC; fixes mistakes with *per-object* rejuvenation

- Non-parametric prior admits a **sequential representation** (right), enabling SMC that incorporates one observation at a time
- Exchangeability of CRPs enables ***per-object*** **rejuvenation moves**: choose any object, and propose new parameters and reference slots — possibly creating new objects of other classes as their targets

2) Fast, data-driven SMC / block rejuvenation proposals created just-in-time by a *proposal compiler*

- Translates subproblem to a Bayes net
- For each pattern of missingness in the data, compiles efficient enumeration-based SMC proposals
- As needed, compiles enumeration-based rejuvenation proposals
- When possible exploits conjugacy for continuous parameters

3) User-specified *inference hints* help scale to large datasets and variable domains

- **subproblem begin … end:**
  Group adjacent statements into a *subproblem* which becomes an intermediate target in SMC. Smaller subproblems lead to more scalable enumerative proposals, at some cost to proposal quality
- **x ~ d(E,…,E) preferring E:**
  Specify a dynamically computed list of values on which posterior is likely to concentrate; proposal enumerates these, and the enumerative proposal is mixed with prior using an adaptive mixture weight

```
GenerateDataset($\Pi$, $\mathbf{Q}$, $|\mathbf{D}|$):
    $\mathbf{R}^{(0)} \leftarrow \emptyset$                    ▷ Initialize empty database
    for observation $i \in \{1, \dots, |\mathbf{D}|\}$ do
        $\Delta_i^{\mathbf{R}} \leftarrow \textsc{GenerateIncrement}(\mathbf{R}^{(i-1)}, C_{obs})$
        $\mathbf{R}^{(i)} \leftarrow \mathbf{R}^{(i-1)} \cup \Delta_i^{\mathbf{R}}$
        $r \leftarrow$ the unique object of class $C_{obs}$ in $\Delta_i^{\mathbf{R}}$
        $d_i \leftarrow \{X \mapsto r.\mathbf{Q}(X),\ \forall X \in \mathcal{A}(\mathbf{D})\}$
    return $\mathbf{R} = \mathbf{R}^{(|\mathbf{D}|)}, \mathbf{D} = (d_1, \dots, d_{|\mathbf{D}|})$
```

```
GenerateDbIncr($\mathbf{R}^{(i-1)}$, root class $C$):
    $\Delta \leftarrow \emptyset$; $r_* \leftarrow$ a new object of class $C$
    for each reference slot $Y \in \mathcal{R}(C)$ do
        $C' \leftarrow T(C.Y)$
        for each object $r \in \mathbf{R}_{C'}^{(i-1)} \cup \Delta_{\mathbf{R}_{C'}}$ do
            $n_r \leftarrow |\{r' \mid r' \in \mathbf{R}^{(i-1)} \cup \Delta \wedge \exists \tau, r'.\tau = r\}|$
        $r_*.Y \leftarrow r$ w.p. $\propto n_r - d_{C'}$, or $\star$ w.p. $\propto$
$s_{C'} + d_{C'}|\mathbf{R}_{C'}^{(i-1)} \cup \Delta_{\mathbf{R}_{C'}}|$
        if $r_*.Y = \star$ then
            $\Delta' \leftarrow \textsc{GenerateDbIncr}(\mathbf{R}^{(i-1)} \cup \Delta, C')$
            $\Delta \leftarrow \Delta \cup \Delta'$
            $r_*.Y \leftarrow$ the unique $r'$ of class $C'$ in $\Delta'$
    for each $X \in \mathcal{A}(C)$, in topological order do
        $r_*.X \sim \phi_{C.X}(\cdot \mid \{r_*.U\}_{U \in Pa(C.X)})$
    return $\Delta \cup \{r_*\}$
```

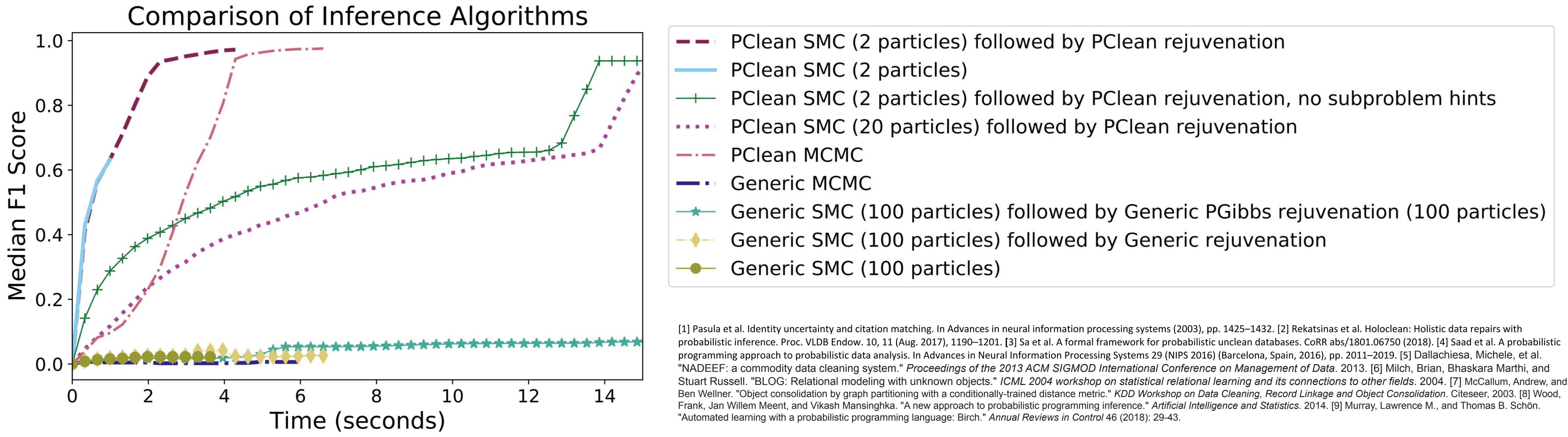
Above: Unlike in general open-universe PPLs, PClean's model admits a **sequential representation**, in which one observation (and all *new* latent objects it is based on) is instantiated at each time step, enabling sequential Monte Carlo inference

4. Experiments & Results

PClean is expressive enough to model data cleaning benchmarks in <50 LoC, and unlike generic PPL inference (left), achieves SOTA accuracy + runtime compared to weighted logic + machine learning baselines.

Task	Metric	PClean	HoloClean (Unpublished)	HoloClean	NADEEF	NADEEF + Manual Java Heuristics
Flights	F_1	**0.90**	0.64	0.41	0.07	**0.90**
	Time	**3.1s**	45.4s	32.6s	9.1s	14.5s
Hospital	F_1	**0.91**	0.90	0.83	0.84	0.84
	Time	**4.5s**	1m 10s	1m 32s	27.6s	22.8s
Rents	F_1	**0.69**	0.48	0.48	0	0.51
	Time	1m 20s	20m 16s	13m 43s	13s	**7.2s**

Table 1: Results of PClean and various baseline systems on three diverse cleaning tasks.



Comparison of Inference Algorithms

[1] Pasula et al. Identity uncertainty and citation matching. In Advances in neural information processing systems (2003), pp. 1425–1432. [2] Rekatsinas et al. Holoclean: Holistic data repairs with probabilistic inference. Proc. VLDB Endow. 10, 11 (Aug. 2017), 1190–1201. [3] Sa et al. A formal framework for probabilistic unclean databases. CoRR abs/1801.06750 (2018). [4] Saad et al. A probabilistic programming approach to probabilistic data analysis. In Advances in Neural Information Processing Systems 29 (NIPS 2016) (Barcelona, Spain, 2016), pp. 2011–2019. [5] Dallachiesa, Michele, et al. "NADEEF: a commodity data cleaning system." *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. 2013. [6] Milch, Brian, Bhaskara Marthi, and Stuart Russell. "BLOG: Relational modeling with unknown objects." *ICML 2004 workshop on statistical relational learning and its connections to other fields*. 2004. [7] McCallum, Andrew, and Ben Wellner. "Object consolidation by graph partitioning with a conditionally-trained distance metric." *KDD Workshop on Data Cleaning, Record Linkage and Object Consolidation*. Citeseer, 2003. [8] Wood, Frank, Jan Willem Meent, and Vikash Mansinghka. "A new approach to probabilistic programming inference." *Artificial Intelligence and Statistics*. 2014. [9] Murray, Lawrence M., and Thomas B. Schön. "Automated learning with a probabilistic programming language: Birch." *Annual Reviews in Control* 46 (2018): 29-43.