Collection-based Operators for Megamodels

Presentor: Nick Fung

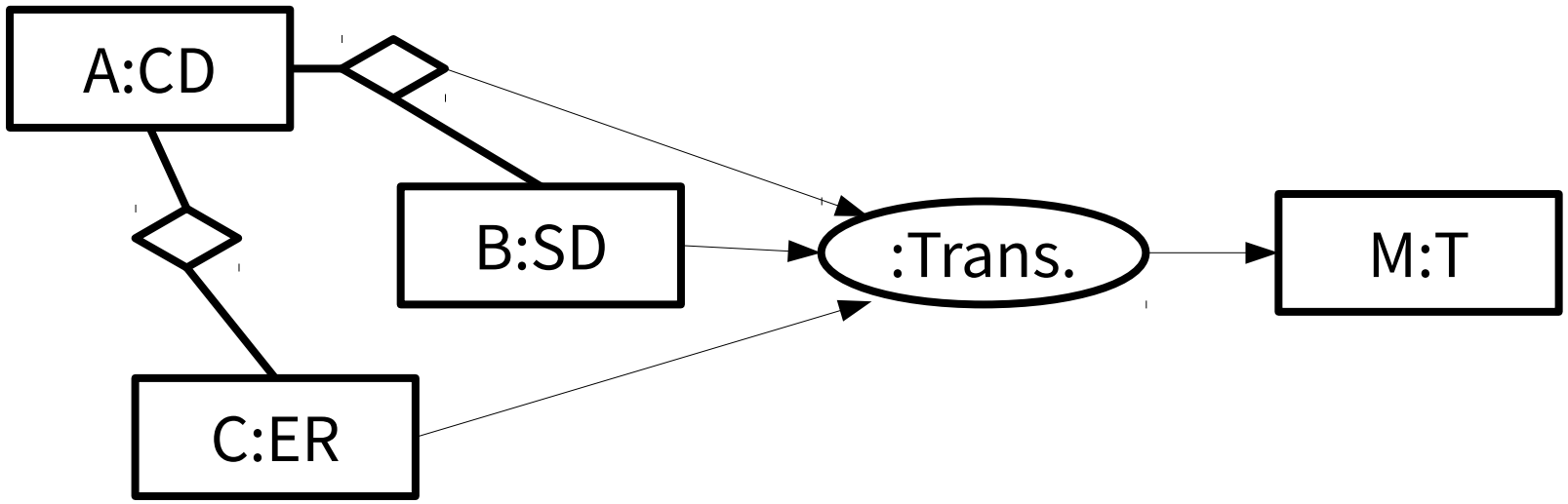
February 26, 2018

Outline

- **Introduction**
- **The Collection-based Operators**
  - Map
  - Reduce
  - Filter
- **The MMINT Workbench**
- **Results**
- **Conclusions**

Megamodels

- **Definition**
  - Collection of related models
- **Mgraph Representation**
  - Models, Relationships, Transformation Apps
  - Megamodels, Megarels, Transformation MegaApps



Collection-based Operators

- **Map**
  - Transforms elements
  - E.g. **`map`**`[λx.2x]([1,2,3,4]) = [2,4,6,8]`
- **Reduce**
  - Aggregates elements
  - E.g. **`reduce`**`[λx,y.x+y]([1,2,3,4]) = 10`
- **Filter**
  - Extracts elements
  - E.g. **`filter`**`[λx.x>2]([1,2,3,4]) = [3,4]`

Map

(Recall: `map[λx.2x]([1,2,3,4]) = [2,4,6,8])`

Input: transformation F with signature $\langle I, O \rangle$,
megamodels $\{X_e | e \in I\}$
Output: set of megamodels $\{Y_e | e \in O\}$

1: **for** $(e \in O)$ { **let** $Y_e := \emptyset$ }
2: **for** (fresh binding K in $\{X_e | e \in I\}$) {
3: **if** F is `commutative` **then**
4: **if** isomorphism of K already done **then continue**;
5: **for** $(e \in O)$ { add element e of $F(K)$ to Y_e } }
6: **return** $\{Y_e | e \in O\}$

Reduce

(Recall: `reduce[λx,y.x+y]([1,2,3,4]) = 10`)

Input: transformation F with signature $\langle I, O \rangle$,
megamodel X
Output: megamodel Y

1: **let** $Y := X$
2: **for** (binding K in Y) {
3: apply $F(K)$ generating output K';
4: **for** $(m \in K_{\texttt{Mod}}, m' \in K'_{\texttt{Mod}}, r(m, m') \in K'_{\texttt{Rel}})$ {
5: **for** $(m'' \in Y_{\texttt{Mod}}, r'(m'', m) \in Y_{\texttt{Rel}})$ {
6: **let** $comp := \texttt{getCompOp}(\texttt{type}(r'), \texttt{type}(r))$;
7: **let** $r''(m', m'') := comp(r', r)$;
8: add r'' to Y }}
9: delete elements in K from Y }
10: **return** Y

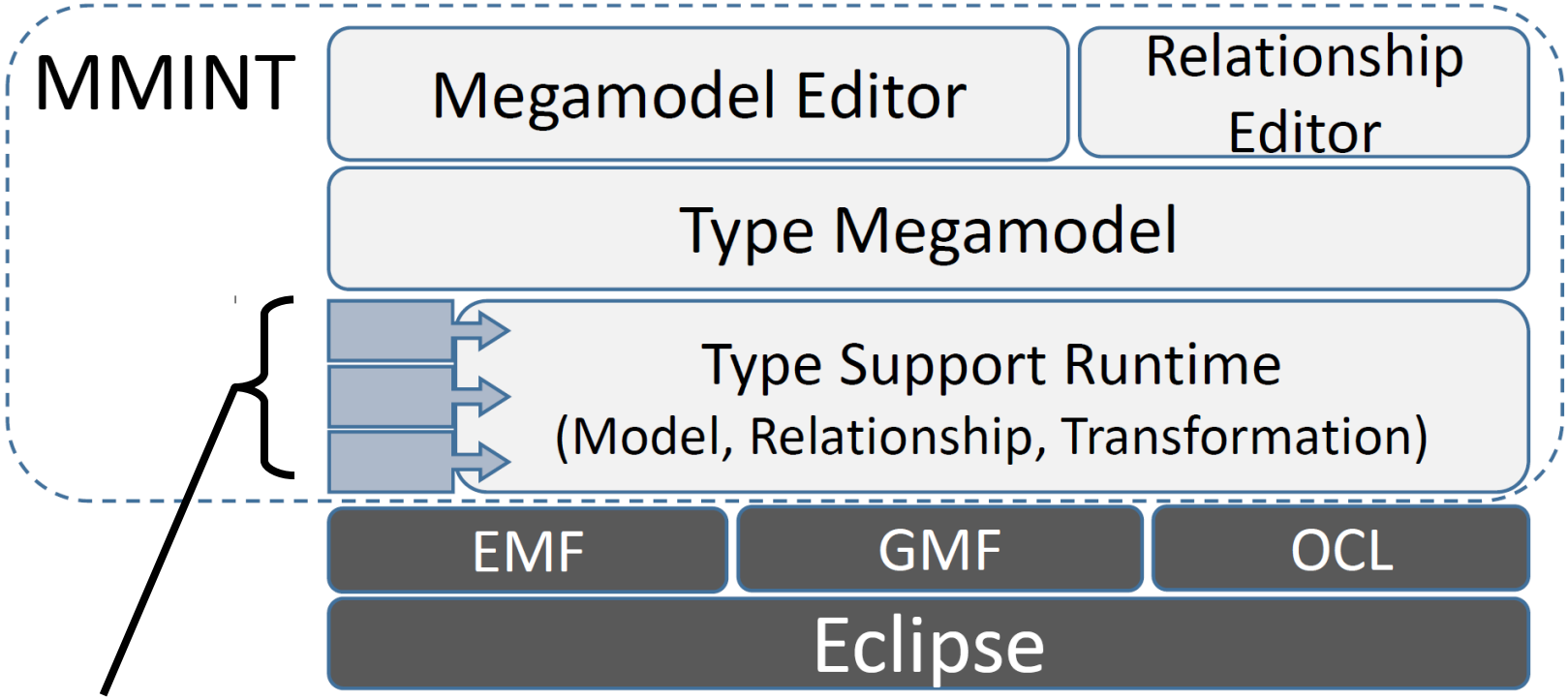
Filter

(Recall: `filter[λx.x>2]([1,2,3,4]) = [3,4]`)

Input: property P, megamodel X
Output: megamodel Y

1: **let** $Y := \emptyset$;
2: **for** $(m \in X_{\texttt{Mod}})$ {
3: **if** P is a model property **then**
4: **if** $m \models P$ **then** add m to Y;
5: **else** add m to Y }
6: **for** $(r \in X_{\texttt{Rel}})$ {
7: **if** P is a relationship property **then**
8: **if** $r \models P$ **then** add r to Y;
9: **else if** $r.\texttt{end} \cap Y \neq \emptyset$ **then** add r to Y }
10: **return** Y;

MMINT Workbench



Metamodels,
Model transformations,
Model editors

Results

- Experiments on Map

	# CDs	# rels	time (sec)	MID size (MB)
exp1	10	100	0.15	0.2
exp2	100	10000	12.92	20.7
exp3	250	62500	85.74	128.9
exp4	500	250000	422.78	518.7

- Complexity Analysis

$\mathbf{map}[F](\{X\})$	$O(n^k \times C_F(m))$
$\mathbf{reduce}[F](X)$	$O(n^2 \times C_F(m))$
$\mathbf{filter}[P](X)$	$O(n^q \times C_P(m))$

Open Questions

- **How can we improve the scalability of the operators?**
  - E.g. Using extra meta-data about the megamodels
- **What other megamodel operators might be useful?**
  - E.g. `while P(MM) do f(MM)`

Summary

- **Megamodels**
  - Mgraph representation
- **Collection-based Operators**
  - Map, Reduce and Filter
- **Open Questions**
  - Scalability improvements
  - Other useful operators