**TRINITY: An Extensible Synthesis Framework for Data Science**

Ruben Martins  
Carnegie Mellon University  
rubenm@andrew.cmu.edu

Jia Chen  
UT Austin  
jchen@cs.utexas.edu

Yanju Chen  
UCSB  
yanjuchen@cs.ucsb.edu

Yu Feng  
UCSB  
yufeng@cs.ucsb.edu

Isil Dillig  
UT Austin  
isil@cs.utexas.edu

---

**Usability**  
TRINITY can be easily adapted to new application domains by providing a suitable DSL (and its semantics) for the target application scenarios.

**Efficient Synthesis**  
TRINITY is based on an efficient synthesis algorithm that combines search and lightweight deduction.

**Customizability**  
TRINITY gives users fine-grained control over inductive bias by providing preference predicates that constrain the space of synthesized programs.

---

**Demo 1: Data Wrangling in R**  
Given only this simple input-output example, TRINITY can automatically synthesize the corresponding R program using the tidyr library:

```r
df1 = gather(input, Score, Value, Score1, Score2)
df2 = unite(df1, AllScores, Grade, Score)
output = spread(df2, AllScores, Value)
```

---

**Demo 2: Extending the DSL for Data Wrangling**  
TRINITY is a modular synthesizer that allows the user to extend existing synthesizers or create new ones. The user can extend the synthesizer by adding this new summarize function to the existing DSL.

---

**Demo 3: Modifying the Search Engine**  
Expert users can also customize the underlying search engine of TRINITY to further speed up their synthesizer. In particular, users can provide statistical models (e.g., deep neural network, n-gram) that can be used to predict the most likely programs.