An Automated Constraint Modelling and Solving Toolchain

Bilal Syed Hussain
Ozgur Akgun, Alan M. Frisch,
Ian Gent, Christopher Jefferson,
Lars Kotthoff, Ian Miguel, Peter Nightingale

Constraint Programming (CP)?

- A powerful technique for solving a wide range of combinatorial problems.
- A constraint satisfaction problem (CSP) is:
  - A set of variables with their associated domains.
  - A set of constraints which specify restrictions on the values the variables can take.

Constraint Modelling

- A constraint model is description of a given problem in terms of CSP.
- There are typically many possible for a given problem.
- Non-experts have difficulty in formulating good constraint models.
- Our approach to solve this difficulty:
  - Refine high level specifications of problems into constraint models.

Automatic constraint modelling

- Our approach to solve this difficulty:
  - Refine high level specifications of problems in our language Essence into constraint models.
  - Allow the user to work solely using high level specifications.
  - Abstracts away the complexity of representation selection and interpreting the results.
  - Retains the efficiency of the constraints solver(s).

Essence

- A high level problem specification language using familiar mathematical notation:
  - Functions
  - Relations.
  - Partitions
  - Sets & mutisets
- Provides nested types:
  - Such as set of set of int

Example: Social Golfers

- In a golf club there are a number of golfers who wish to play together in **g** groups of size **s**.
- Find a schedule of play for **w** weeks such that no pair of golfers play together more than once.

Social Golfers: Modelling

- In each week, we need to **partition** the golfers into groups.
- What about the weeks?
  - A sequence? But what does the order matter?
  - A multiset then.
- So the problem is to find a **multiset** of **partitions**.

Social Golfer: Essence

- In a golf club there are a number of golfers who wish to play together in **g** groups of size **s**.

```
given w, g, s : int(1..)
```

- This is a class of problems, to get an instance parameters have to be provided.

Social Golfer: Essence

- There are g × s golfers, which are interchangeable.

```
letting Golfers be new type of size g * s
```

- Find a schedule of play for **w** weeks.

```
find sched : mset (size w) of
  partition (regular, size g) from Golfers
```

Social Golfer: Socialisation Constriant

- such that no pair of golfers play together more than once.

```
such that
forAll week1 in sched .
 forAll week2 in sched , week1 != week2 .
```

For each pair of distinct weeks,

Social Golfer: Socialisation Constraint

```
such that
forAll week1 in sched .
 forAll week2 in sched , week1 != week2 .
    forAll group1 in parts(week1) .
      forAll group2 in parts(week2) .
        |group1 intersect group2| < 2
```

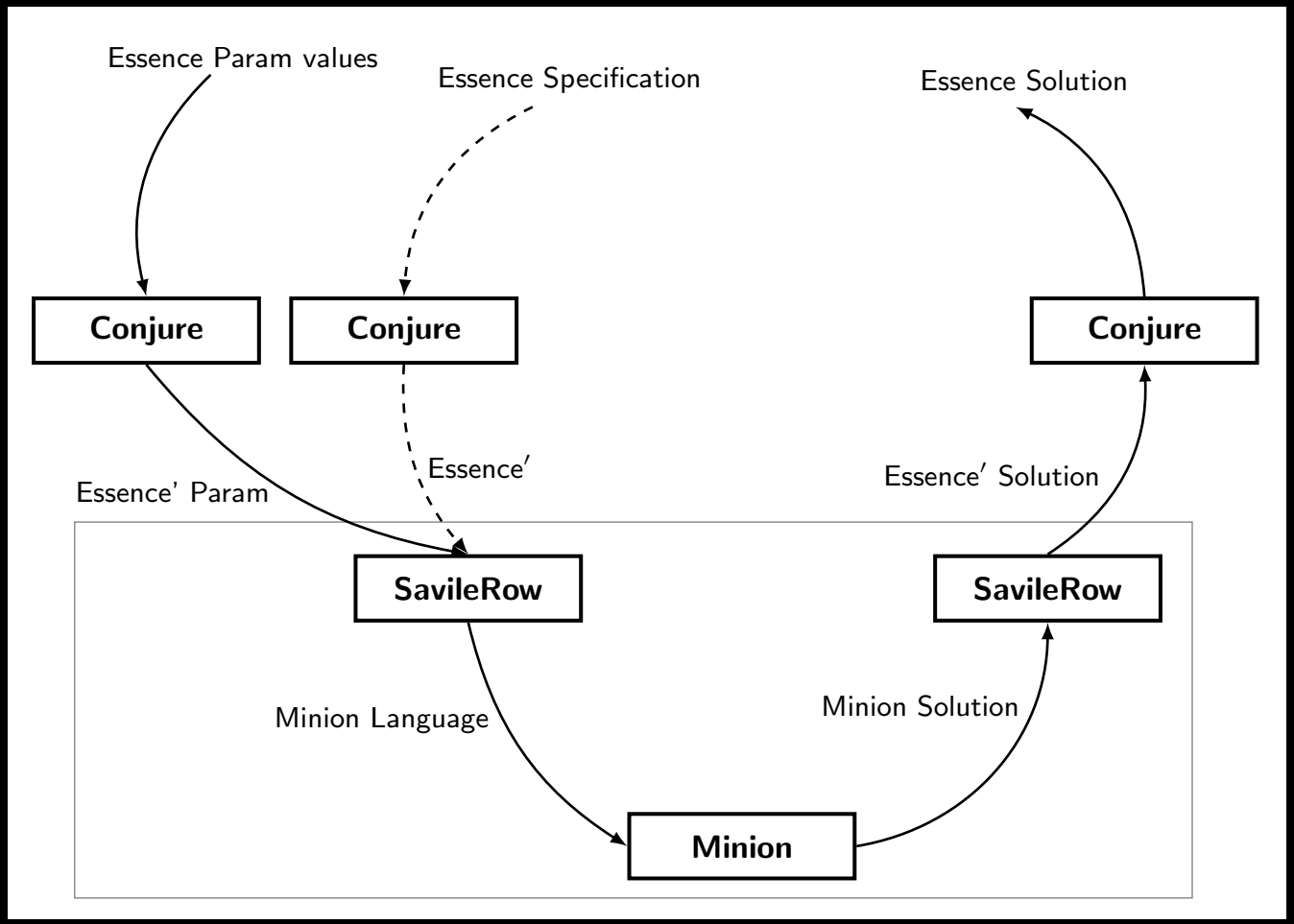
The intersection of any pair of groups is at most one golfer.

Social Golfer: Complete Essence

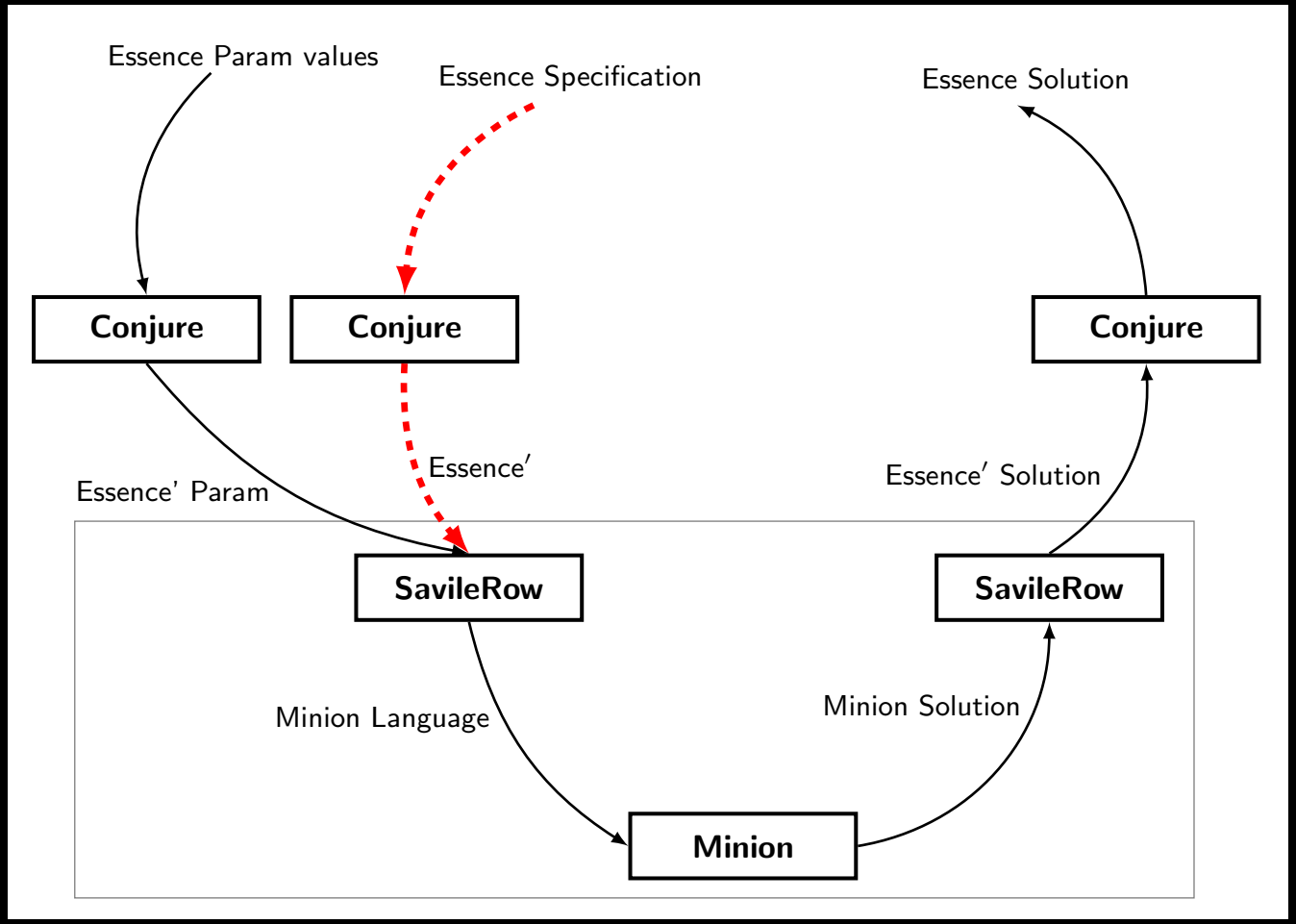
```
language Essence 1.3.0
given w, g, s : int(1..)
letting Golfers be new type of size g * s
find sched : mset (size w) of
 partition (regular, size g) from Golfers

such that
forAll week1 in sched .
 forAll week2 in sched , week1 != week2 .
    forAll group1 in parts(week1) .
      forAll group2 in parts(week2) .
        |group1 intersect group2| < 2
```

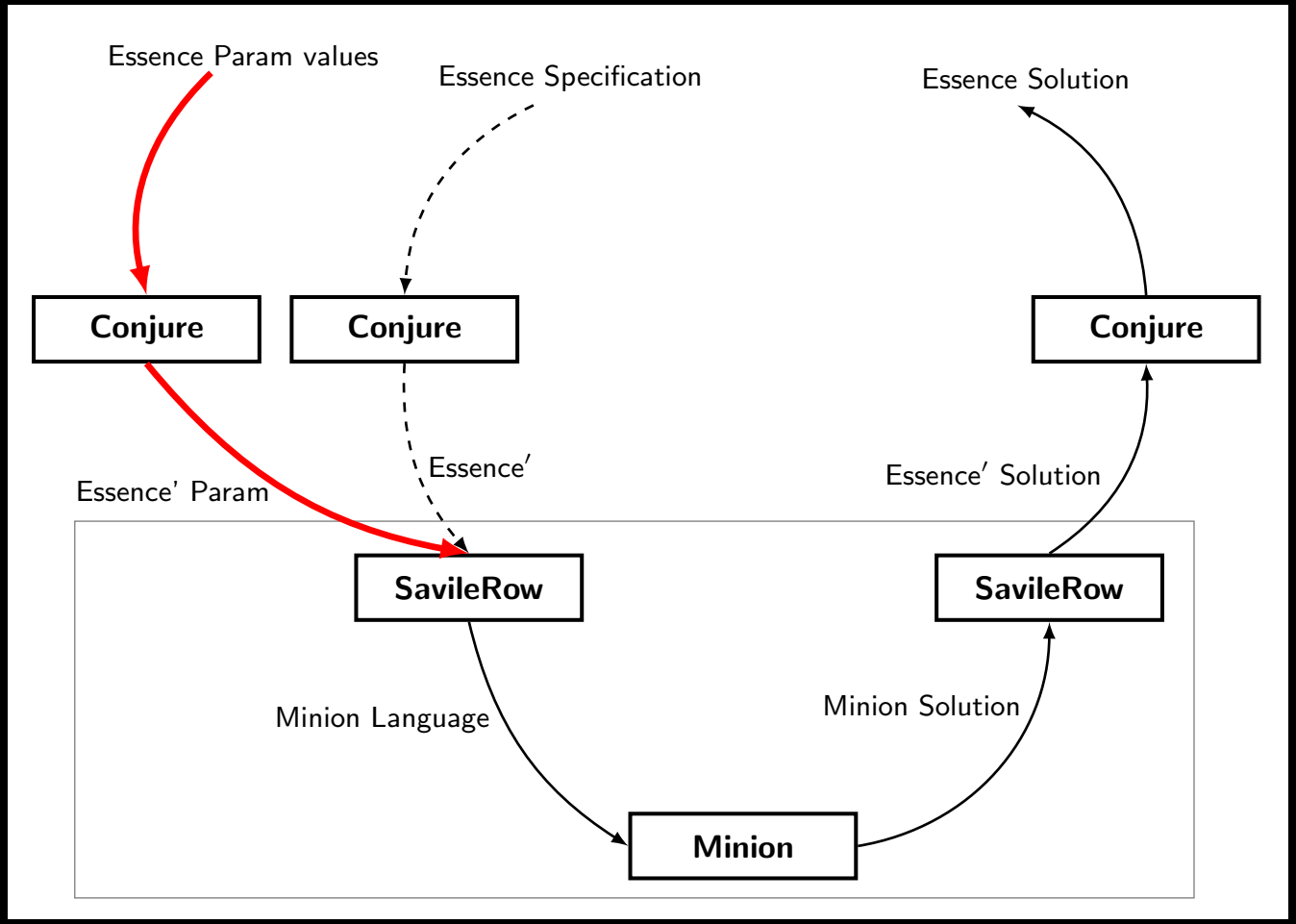
An overview of the Toolchain



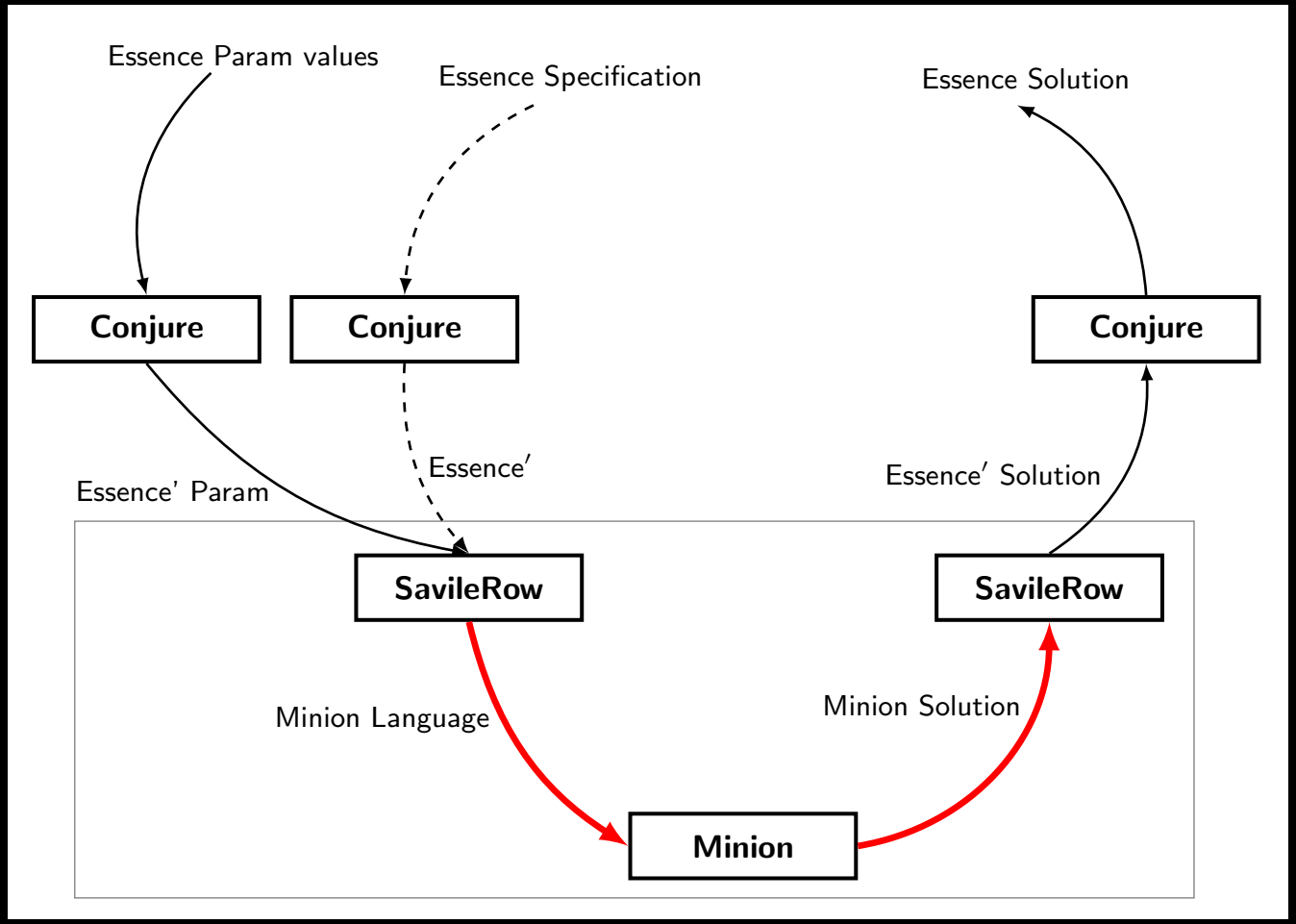
An overview of the Toolchain



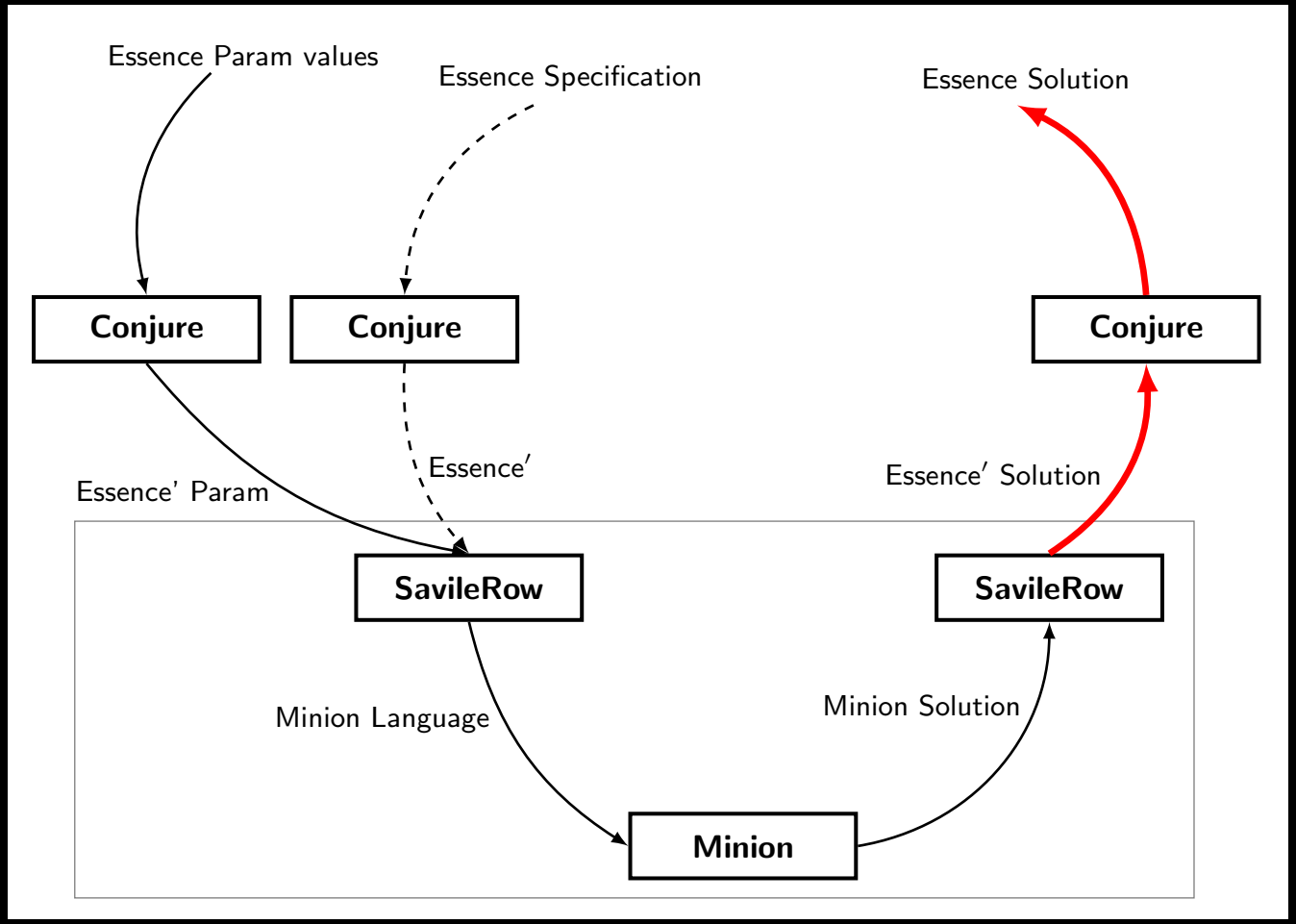
An overview of the Toolchain



An overview of the Toolchain



An overview of the Toolchain



An overview of the Toolchain

- Conjure: Generates models from the problem specification, Essence $\rightsquigarrow$ Essence'.
- SavileRow: Compiles highly efficient models, Essence' $\rightsquigarrow$ Minion.
- Minion: A very efficient CP solver.

Model Selection

- Conjure can generate multiple models for a given problem.
- Having multiple models is: *an opportunity rather than a difficulty.*
- Conjure can pick all the generated models, or a selection by:
  - Randomly picking.
  - Using heuristics.

Conclusion

- Make CP more accessible.
  - allowing the user to write a high level specifications while retaining the high performance of the CP solvers.
- Allows the use of nested types and familiar mathematical notation.
- No expertise needed:
  - To model the problem.
  - To select a good model.