We can compute the FF heuristic directly from a PDDL description and achieve state-of-the-art performance among lifted planners.

## Relaxed Plans Using Annotated Datalog Programs

We can compute the FF heuristic directly from a PDDL description and achieve state-of-the-art performance among lifted planners.

```datalog
(:action A
 :parameters (?X ?Y)
 :precondition
 (and (P ?X ?Y) (S ?X))
 :effect
 (and (Q ?X) (R ?Y)))

A-app(X,Y) :- P(X, Y), S(X) [Add A(X, Y) to the plan]
Q(X) :- A-app(X,Y) [ ]
R(Y) :- A-app(X,Y) [ ]
goal() :- Q(0). [ ]
```

### Annotations
- Sequence of instructions
- Ground with the rule
- Executed after deriving goal()
- Backchain through achievers

### Transformations
Direct encoding does not scale.

We need to apply transformations:
- Rule Splitting
- Rule Merging
- Predicate Collapsing
- Variable Renaming

Semantics preserved with transformations

### Table

<table>
<thead>
<tr>
<th>Heuristics</th>
<th>IPC</th>
<th>HTG</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifted FF + Lazy P.O.</td>
<td>782</td>
<td>691</td>
<td>1473</td>
</tr>
<tr>
<td>Lifted Additive + Lazy P.O.</td>
<td>762</td>
<td>663</td>
<td>1425</td>
</tr>
<tr>
<td>Lifted Goalcount + Unary Relax.</td>
<td>575</td>
<td>641</td>
<td>1216</td>
</tr>
<tr>
<td>Ground FF + Lazy P.O.</td>
<td>862</td>
<td>595</td>
<td>1457</td>
</tr>
</tbody>
</table>

### Example

Justification Tree

```
goal()
   \|-- Q(0)
       \|-- A-app(0,1)
            \|-- P(0,1)
                 \|-- S(0)
```

Order annotations bottom-up and execute instructions to compute FF