Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»
КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчёт по защите ЛР №10 по дисциплине «Функциональное и логическое программирование»

Тема: Библиотека clpfd
Студент: Карпова Е. О., Княжев А. В.
Группа: ИУ7-62Б
Оценка (баллы): _____
Преподаватели: Толпинская Н. Б., Строганов Ю. В.

Москва — 2023 г.

1. Теоретическая часть

1.1. Библиотека clpfd. Логическое программирование в конечных доменах

2 основных варианта использования clpfd:

1. декларативная арифметика целых чисел;
2. решение комбинаторных задач, таких как планирование, диспетчеризация и распределение задач.

Предикаты этой библиотеки могут классифицироваться как:

1. арифметические ограничения `#=`, `#>`, `#\=`;
2. ограничения членства `in` и `ins`;
3. предикаты перечисления `indomain`;
4. комбинаторные ограничения `all_distinct`;
5. комбинаторные ограничения `<==>`;
6. комбинаторные ограничения `fd_dom`.

Для подключения библиотеки нужно добавить строчку:

Листинг 1..1: Подключение

```
1 :- use_module(library(clpfd)).
```

1.2. Арифметические ограничения

1.2.1. Эквивалентность

Одним из наиболее важных ограничений является ограничение эквивалетности `#=`. Оно ведет себя как `is` и `=:=` над целыми числами, кроме того, работает в обоих направлениях.

Листинг 1..2: Пример эквивалетности

```
1  ?- X #= 1+2.
2  X = 3.
3
4  ?- 3 #= Y+2.
5  Y = 1.
6
7  ?- 3 is Y+2.
8  ERROR: is/2: Arguments are not sufficiently instantiated
9
10 ?- 3 =:= Y+2.
11 ERROR: =:=/2: Arguments are not sufficiently instantiated
```

В примере ниже с использованием `#=` реализован факториал. Оно позволяет производить вычисления факториала в обе стороны:

Листинг 1..3: Реализация факториала

```
1  fact1(0, Acc, Acc) :- !.
2  fact1(N, Acc, Res) :- N #> 0, Acc #=< Res, Res #> 0, Acc1 #= N * Acc, N1
       #= N - 1, fact1(N1, Acc1, Res).
3
4  fact(N, Res) :- fact1(N, 1, Res).
```

Листинг 1..4: Примеры запросов

```
1  ?- fact(5, N).
2  N = 120.
3
4  ?- fact(N, 120).
5  N = 5.
6
7  ?- fact(N, 11).
8  false.
```

1.2.2. Другие ограничения

1. `Expr1 #= Expr2` — Expr1 равно Expr;

2. `Expr1 #\= Expr2` — Expr1 не равно Expr;

3. `Expr1 #>= Expr2` — Expr1 больше или равно Expr2;

4. `Expr1 #=< Expr2` — Expr1 меньше или равно Expr2;

5. `Expr1 #> Expr2` — Expr1 больше Expr2;

6. `Expr1 #< Expr2` — Expr1 меньше Expr2.

Expr1 и Expr2 — арифметические выражения, которые могут быть либо целыми числами, либо результатом арифметических операций, таких как `+, -, *, ^, min, max, abs, mod, rem, //, div`, совершенных над арифметическими выражениями.

1.3. Комбинаторные ограничения

1. `all_distinct` — верно, если все элементы аргумента попарно различны;

2. `global_cardinality` — верно, если для каждого ключа второго аргумента-словаря, значение равно количеству вхождений этого ключа в первый аргумент;

3. `cumulative` — описывает расписание фиксированного ресурса.