

1 Classical Planning Overview

1.1 Background

1.2 PDDL

1.3 FF Planner

1.4 Planning Domain Examples

1.4.1 Blocks World

Listing 1: Blocks World Domain Description in PDDL

```
1 (define (domain BLOCKS)
2   (:requirements :strips :typing)
3   (:types block)
4   (:predicates (on ?x - block ?y - block)
5                 (ontable ?x - block)
6                 (clear ?x - block)
7                 (handempty)
8                 (holding ?x - block)
9                 )
10
11  (:action pick-up
12    :parameters (?x - block)
13    :precondition (and (clear ?x) (ontable ?x) (handempty))
14    :effect
15    (and (not (ontable ?x))
16         (not (clear ?x))
17         (not (handempty))
18         (holding ?x)))
19
20  (:action put-down
21    :parameters (?x - block)
22    :precondition (holding ?x)
23    :effect
24    (and (not (holding ?x))
25         (clear ?x)
26         (handempty)
27         (ontable ?x)))
28  (:action stack
29    :parameters (?x - block ?y - block)
30    :precondition (and (holding ?x) (clear ?y))
31    :effect
32    (and (not (holding ?x))
33         (not (clear ?y))
34         (clear ?x)
35         (handempty)
36         (on ?x ?y)))
37  (:action unstack
38    :parameters (?x - block ?y - block)
39    :precondition (and (on ?x ?y) (clear ?x) (handempty))
40    :effect
41    (and (holding ?x)
42         (clear ?y)
43         (not (clear ?x))
44         (not (handempty))
45         (not (on ?x ?y))))
```

1.4.2 Towers of Hanoi

Listing 2: Towers of Hanoi Domain Description in PDDL

```
1 ;; towers of hanoi
2
3 (define (domain HANOI)
4   (:requirements :typing)
5   (:types disc peg)
6   (:predicates
7     (clear ?x)
8     (on ?x - disc ?y)
9     (larger ?d - disc ?e - disc)
10  )
11
12  (:action stack-d
13    :parameters (?d - disc ?e - disc)
14    :vars (?l)
15    :precondition (and
16      (on ?d ?l)
17      (not (on ?d ?e))
18      (not (= ?d ?e))
19      (not (= ?e ?l))
20      (larger ?e ?d)
21      (clear ?d)
22      (clear ?e)
23    )
24    :effect (and
25      (not (on ?d ?l))
26      (not (clear ?e))
27      (on ?d ?e)
28      (clear ?l)
29    )
30  )
31
32  (:action stack-p
33    :parameters (?d - disc ?p - peg)
34    :vars (?l)
35    :precondition (and
36      (on ?d ?l)
37      (clear ?p)
38      (clear ?d)
39      (not (= ?p ?l))
40    )
41    :effect (and
42      (not (clear ?p))
43      (not (on ?d ?l))
44      (on ?d ?p)
45      (clear ?l)
46    )
47  )
48 )
```

1.4.3 Lin's Briefcase

Listing 3: Lin's Briefcase Domain Description in PDDL

```
1 ;; briefcase domain
2
3 (define (domain BRIEFCASE)
4   (:requirements :typing)
```

```

5  (:types latch)
6  (:predicates
7    (open)
8    (latched ?l - latch)
9  )
10
11  (:action flip-open
12    :parameters (?l - latch)
13    :precondition (latched ?l)
14    :effect (not (latched ?l))
15  )
16
17  (:action flip-closed
18    :parameters (?l - latch)
19    :precondition (not (latched ?l))
20    :effect (latched ?l)
21  )
22
23  (:action open
24    :parameters ()
25    :precondition (and
26      (forall (?l - latch)
27        (not (latched ?l))
28      )
29      (not (open))
30    )
31    :effect (open)
32  )
33 )

```

1.4.4 Electrical Circuit

Listing 4: Electrical Circuit Domain Description in PDDL

```

1  ;; circuit domain
2
3  (define (domain CIRCUIT)
4    (:requirements :typing :conditional-effects)
5    (:types wire gate level)
6    (:predicates
7      (wire-high ?w - wire)
8      (gate-active ?g - gate)
9      (gate-level ?g - gate ?l - level)
10     (and-gate ?g - gate)
11     (or-gate ?g - gate)
12     (inv-gate ?g - gate)
13     (input-to ?w - wire ?g - gate)
14     (output-from ?w - wire ?g - gate)
15   )
16
17   (:action activate-wire
18     :parameters (?w - wire)
19     :vars (?g2 - gate)
20     :precondition (and
21       (not (wire-high ?w))
22       (input-to ?w ?g2)
23       (forall (?g - gate)
24         (not (output-from ?w ?g))
25       )
26     )
27     :effect (and

```

```

28         (wire-high ?w)
29     )
30 )
31
32 (:action deactivate-wire
33   :parameters (?w - wire)
34   :vars (?g2 - gate)
35   :precondition (and
36     (wire-high ?w)
37     (input-to ?w ?g2)
38     (forall (?g - gate)
39       (not (output-from ?w ?g))
40     )
41   )
42   :effect (and
43     (not (wire-high ?w))
44   )
45 )
46 (:action activate-and-gate
47   :parameters (?g - gate)
48   :vars (?w1 - wire ?w2 - wire ?w3 - wire)
49   :precondition (and
50     (and-gate ?g)
51     (not (gate-active ?g))
52     (input-to ?w1 ?g)
53     (input-to ?w2 ?g)
54     (output-from ?w3 ?g)
55     (wire-high ?w1)
56     (wire-high ?w2)
57     (not (= ?w1 ?w2))
58     (not (= ?w1 ?w3))
59     (not (= ?w2 ?w3))
60   )
61   :effect (and
62     (wire-high ?w3)
63     (gate-active ?g)
64   )
65 )
66
67 (:action activate-inv-gate
68   :parameters (?g - gate)
69   :vars (?w1 - wire ?w2 - wire)
70   :precondition (and
71     (inv-gate ?g)
72     (not (gate-active ?g))
73     (input-to ?w1 ?g)
74     (output-from ?w2 ?g)
75     (not (wire-high ?w1))
76   )
77   :effect (and
78     (wire-high ?w2)
79     (gate-active ?g)
80   )
81 )
82
83 (:action activate-or-gate
84   :parameters (?g - gate)
85   :vars (?w1 - wire ?w2 - wire ?w3 - wire)
86   :precondition (and
87     (or-gate ?g)
88     (not (gate-active ?g))

```

```
89         (input-to ?w1 ?g)
90         (input-to ?w2 ?g)
91         (output-from ?w3 ?g)
92         (or (wire-high ?w1) (wire-high ?w2))
93     )
94     :effect (and
95         (wire-high ?w3)
96         (gate-active ?g)
97     )
98 )
99 )
```