Table 4. Learning static reject rules for the action "UNLOAD-AIRPLANE (o p a)". The types of the variables are: o - package; p - airplane; a - airport; c - city; and l - location.

	time	o	p	a	c	l
+	2	o1	pln	apt-A	A	po-C
+	3	o1	pln	apt-B	B	po-C
+	4	o1	pln	apt-B	B	po-C
+	4	o2	pln	apt-B	B	po-C
-	5	o1	pln	apt-C	C	po-C
-	5	o2	pln	apt-C	C	po-C

(o p a)". (See the caption of Table 4 for the types associated with each variable in this example.) According to the heuristic for generating training examples for static reject rules, UNLOAD-AIRPLANE (o1 pln apt-A) at time 2 is a positive example because all of its preconditions hold at time 2 (such as (in o1 pln) and (at pln apt-A)) but it does not appear in the plan. On the other hand, UNLOAD-AIRPLANE (o1 pln apt-C) at time 5 is a negative example because it appears in the plan at time 5. The complete set of positive and negative examples are shown in the first five columns in Table 4.

Following the learning procedure outlined above, two determinate literals (in-city a c) and (goal (at o l)) are first added to the rule, and they introduce two new variables c and l (see last two columns in Table 4).

In the next iteration, a literal ¬(in-city l c) is found to give the highest possible gain and is added to the rule. Now the rule covers only positive examples and none of negative examples. Therefore the procedure terminates with the rule:

¬ UNLOAD-AIRPLANE (o p a) ←
(in-city a c) ∧ (goal (at o l)) ∧ ¬(in-city l c)

It is worth noting that the above rule captures the concept of "an object that is not in its goal city". In other recent work on learning control knowledge for planning (Estlin & Mooney, 1996), this concept can only be learned by introducing hand-coded background knowledge.

4. Experimental Results

We have performed a preliminary empirical evaluation of our approach on a set of planning domains (logistics, grid, gripper, and mystery) from the 1998 AI Planning Systems Competition (AIPS98), as well as the tireworld domain from the PDDL (McDermott, 1998) and TLPlan (Bacchus & Kabanza, 2000) distributions. All experiments were run on a 300Mhz Sparc Ultra. As noted earlier, the logistics domain has become a particularly popular benchmark for recent work in planning.

Table 5. Learning time (in seconds) and number of rules acquired. Learning time includes time to both generate and verify rules. Mystery training problems are from AIPS98 competition, and Tireworld problems are from PDDL distribution. All other training problems are randomly generated.

domain	# training problems	learning time	# rules learned
grid	6	66.79	10
gripper	2	0.13	3
logistics	10	22.54	11
mystery	6	120.6	4
tireworld	4	1.23	17

Table 6. Blackbox without and with (c) learned control knowledge. Results are averaged over 10 runs and times are given in cpu seconds. Grid and tireworld problems are randomly generated. Logistics-d and logistics-e are from the Blackbox distribution. All other problems are from the AIPS98 competition.

problem	time step (# obj/goal)	Blackbox	Blackbox(c)
grid-a	13 (28/3)	20.99	4.81
grid-b	18 (29/1)	74	16.62
grid-c	23 (29/2)	2132	37.03
gripper-p03	15 (12/8)	> 7200	7.18
gripper-p04	19 (14/10)	> 7200	259.8
logistics-d	14 (36/9)	15.85	5.76
logistics-e	15 (40/10)	3522	290.9
logistics-p05	12 (43/4)	> 7200	10.58
logistics-p07	9 (60/6)	1522	32.34
mystery-p10	8 (77/1)	> 7200	47.25
mystery-p13	8 (83/2)	161	12.24
tireworld-a	24 (15/19)	5.8	3.65
tireworld-b	30 (17/23)	60.6	28.39

Table 5 summarizes the learning time and number of control rules acquired for each domain. Most of the training examples are randomly generated small instances, while a few are taken directly from the available distributions. In general our learning times are very short relative to other speed-up learning systems, which typically take several minutes to several hours to generate a good set of rules.

Table 6 summarizes the results of running Blackbox with and without the learned control rules on a set of larger and harder problems drawn from each domain. For each domain Blackbox's various parameters are set to optimize performance for the case where no learned rules are added. To give a very rough idea of the size of the each instance, the solution parallel length (time steps), number of objects (constants), and number of