Goal Stack Planner Report

Description:

Goal:

Find operators that change an initial state of a game to goal state. The game involves objects, and operators to manipulate these objects. The objects relationships between other objects define the game state.

How:

Block worlds used as game.
Objects are the blocks, hand (picking up blocks) and table
Relationship between objects are: block above block, block below block, block above table, block in hand.
Two operators, pick up and put down.

In a game multiple goals define the goal state.

Each unsatisfied goal attempted to be satisfied concurrently. The first sequence of operators to satisfy a goal is operated on game state.
Repeated game states are not allowed.

Repeated until the game's goal state is reached.

Game:

Goal:

Store objects, objects' relationships with each other, goal state and operators to manipulate the objects, report which goals are satisfied and if goal state is reached.

Handler:

Goal:
Find operators that change an initial state of a game to goal state.

How:
Legal operators(operators that can be applied in current game state) and illegal operators(operators that can not be applied in current game state) are put into separate lists.

A thread is started for each unsatisfied goal state.

Each thread searches backwards from an operator that would satisfy its goal(usually illegal operator). The shortest sequence of operators that if applied would make this operator legal is found.

If the first operator legal then this sequence of operators can be applied.
If the first operator is illegal the process must be repeated for for this first operator. Repeated until the first operator in the sequence is legal.

To stop circular motion between game states(solution would never be found) the game state, after applying a sequence of operators, can not be repeated.

The fastest thread to find its solution to its goal is operated on the game state. Other threads are killed.

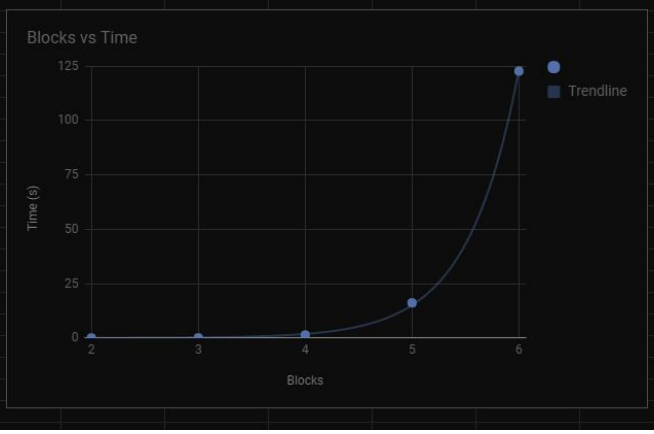
This method is repeated until the game's goal state is reached.

Efficiency:

Found solutions for games of up to 6 objects.
Complexity of algorithm is exponential, as shown below.

Blocks	Time	Games tested
2	0.00591403	10000
3	0.049199	1000
4	1.355	100
5	16.1528	20
6	122.6813333	3

Complexity exponentioal: e^(2.101x)



Algorithm is ineffective at solving games of even just 7 objects or more due to the explosion of number of sequences of operators that must be considered.

Take for example the blocks world game used to test the Handler algorithm, which has 2 operators and 9 objects for a 7 block game(7 blocks + 1 hand + 1 table).
The number of sequences of operators of 3 operators to consider is
(sequence size)*(number of operators)*(number of 2 pairs of objects)=
(3)*(2)*(9!) = 2,177,280 sequences of operators to consider.

To lessen this explosion of sequences of operators to consider, general solutions to satisfying goals could be cached into a list of an object that contains: initial state, goal state, general operators to achieve this. So before searching exhaustively for a solution the program can check if it has already solved something that can be applied to satisfy this goal.