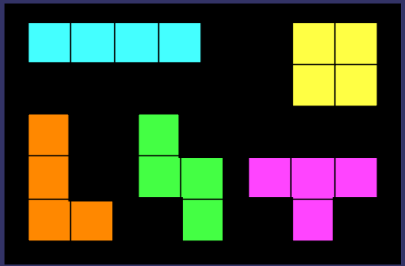
Bin Packing Using Search Methods

Robert Jackson

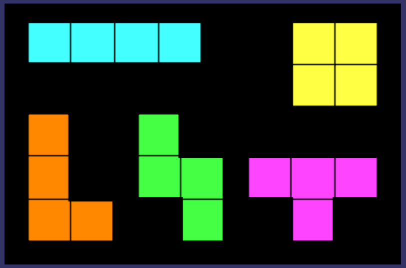
Applications
Current State of the Art
Shapes
Bins
Rotations
Search Methods
Heuristics



Overview

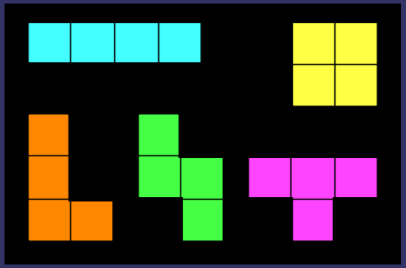
- Bin Packing optimization deals with the problem of fitting objects of various shapes and sizes into a bin of length *l* and width *w.*

- Search algorithms (DFS, A*) can be used to traverse the possibility tree to find a config-uration that leaves no holes.



Long-term goal

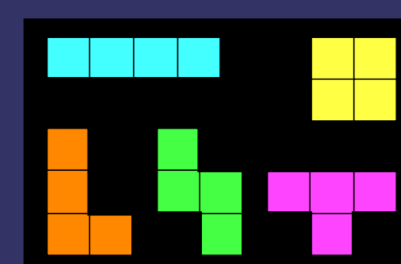
- Given a set of tetrominos T, and a bin of length l and width w, place each element of T at location x,y and rotation r.
- numTetrominos*4 <= l*w
- numTetrominos*4 + numHoles <= l*w



The Problem

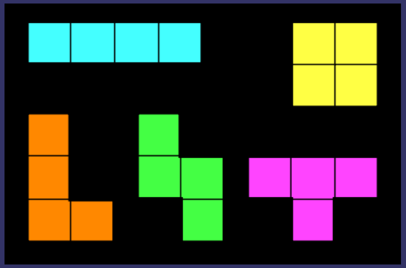
Storage of Radioactive
Waste

Shipping of Goods
Of various size/shapes



The Present Situation

- Give a summary of the current situation
- Packing for:
  - Storage – optimize space, time
  - Transport – inertia, center of mass, space
- Applications
  - Memory Management (Paging)
  - Shipping
  - Storage
  - Art
- Commercial Application Suites
  - optimal packing of various bins
  - shipping containers
  - pallets
  - radioactive waste storage/transport



Development up to present

- Development made up to the current situation
- Important background information
- Original forecasts which turned out to be wrong
- Original forecasts which turned out to be true

Potential Alternatives

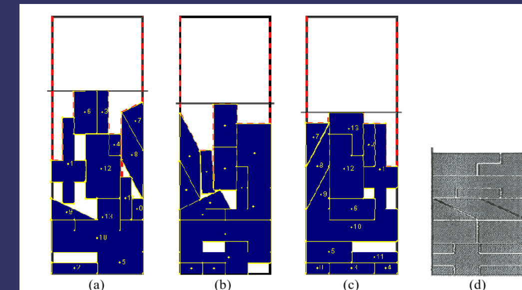
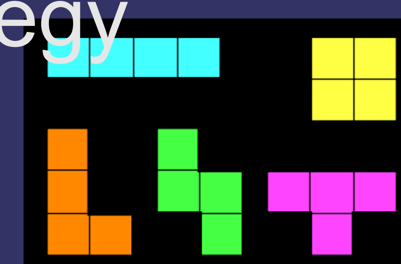


Figure 5-2. Arrangement of 14 geometric patterns (a) 2nd generation, (b) 36th generation, (c) 84th generation, (d) output from Watanabe and Ono (cited in Das97)

- State the alternative strategies
  - Genetic Algorithm (GA)
    - Pros: good for large tree,
    - Cons: slow, not optimal,
  - Constraint Satisfaction Problem (CSP)
    - Pros: fast
    - Cons: not easily modified
  - Packing homogeneous shapes/sizes
    - Pros: greatly reduces permutations
    - Cons: not possible in most instances
- List the pros and cons of each strategy
- Give a forecast of costs



Genetic Algorithm by Arfath Pasha

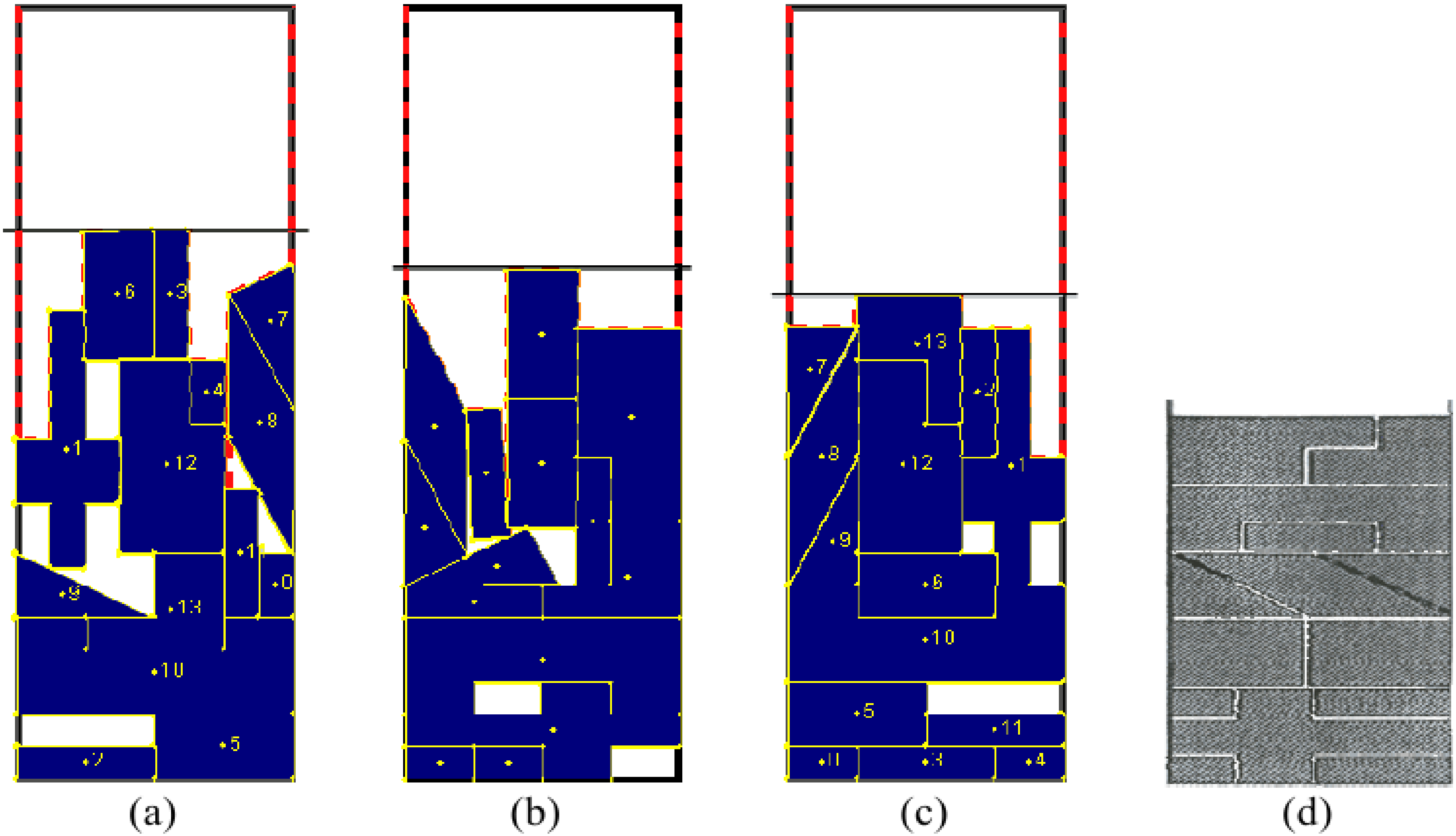
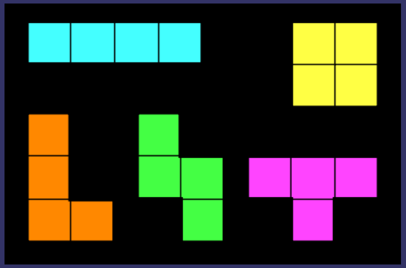


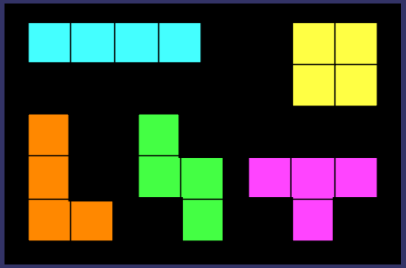
Figure 5-2. Arrangement of 14 geometric patterns (a) 2nd generation, (b) 36th generation, (c) 84th generation, (d) output from Watanabe and Ono (cited in Das97)

- Bin Packing Methods
- Overview
- Long-term goal
- The Present Situation
- Development up to present
- Potential Alternatives
- Recommendation



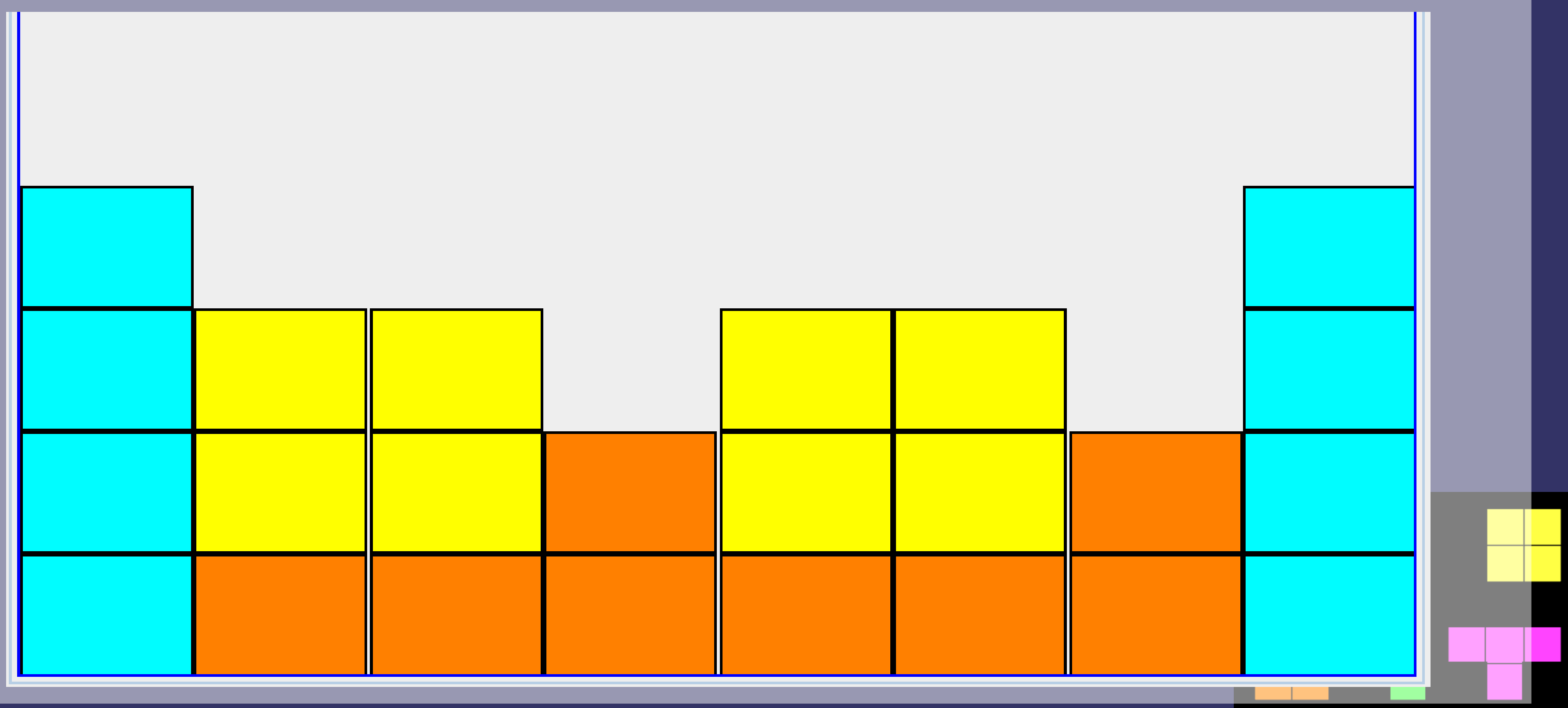
My Project – Tetromino Bin Packer

- Algorithms Implemented
  - A* (NumberOfTilesLeft and NumberOfHoles Heuristics)
  - DSF
  - Greedy Best First Search
  - Hill Climbing
  - Simulated Annealing



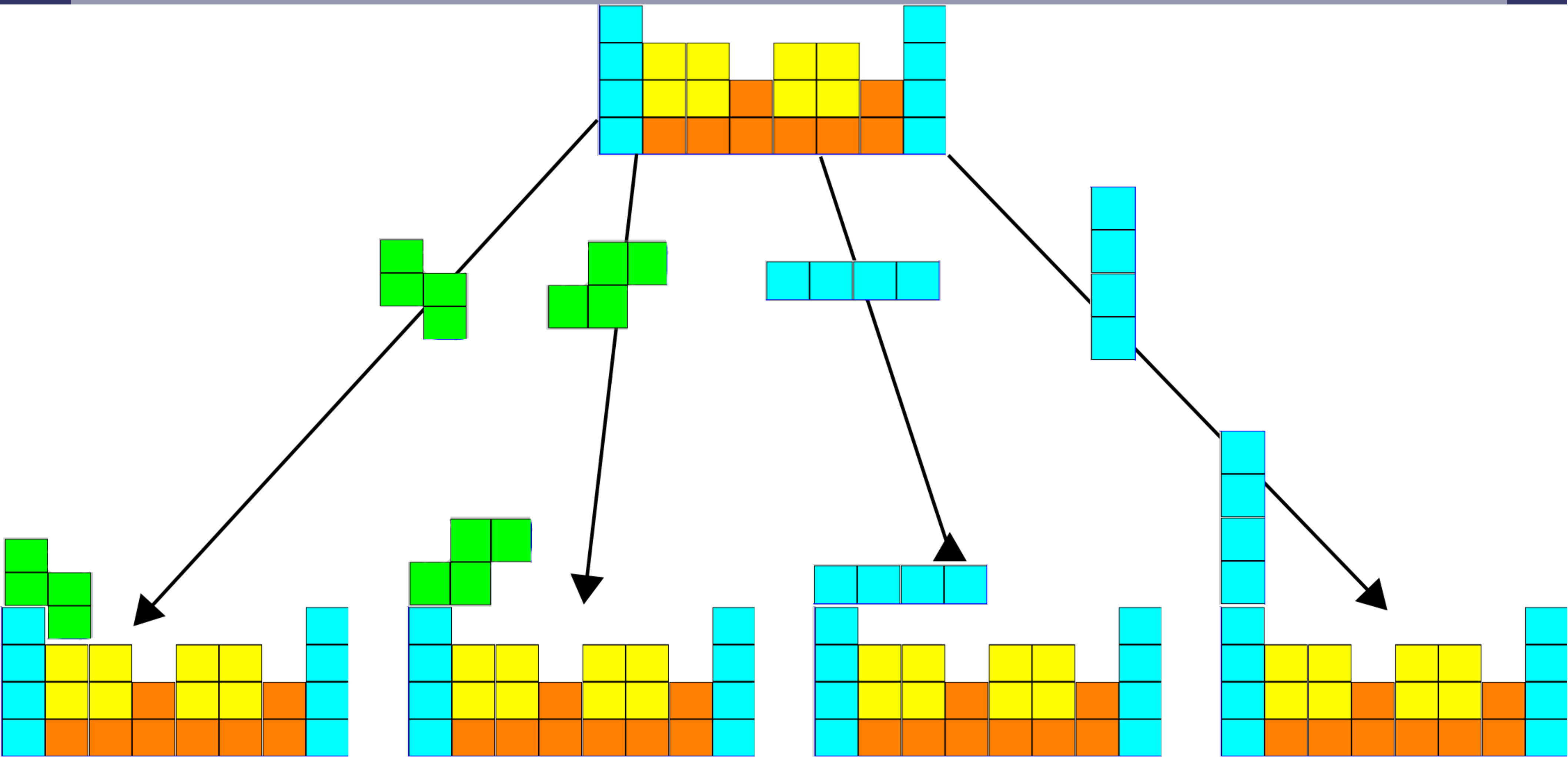
Building the Tree – 1

```
class TetrisPieceActionsFunction {
public Set<Action> actions(Object state) {
…
for (int j = 0; j < board.getRows(); j++) {
     ...
     for (int i = 0; i < board.getColumns(); i++) {
          addActionsToSet(pool, i, j, board, actions);
     }
}
```



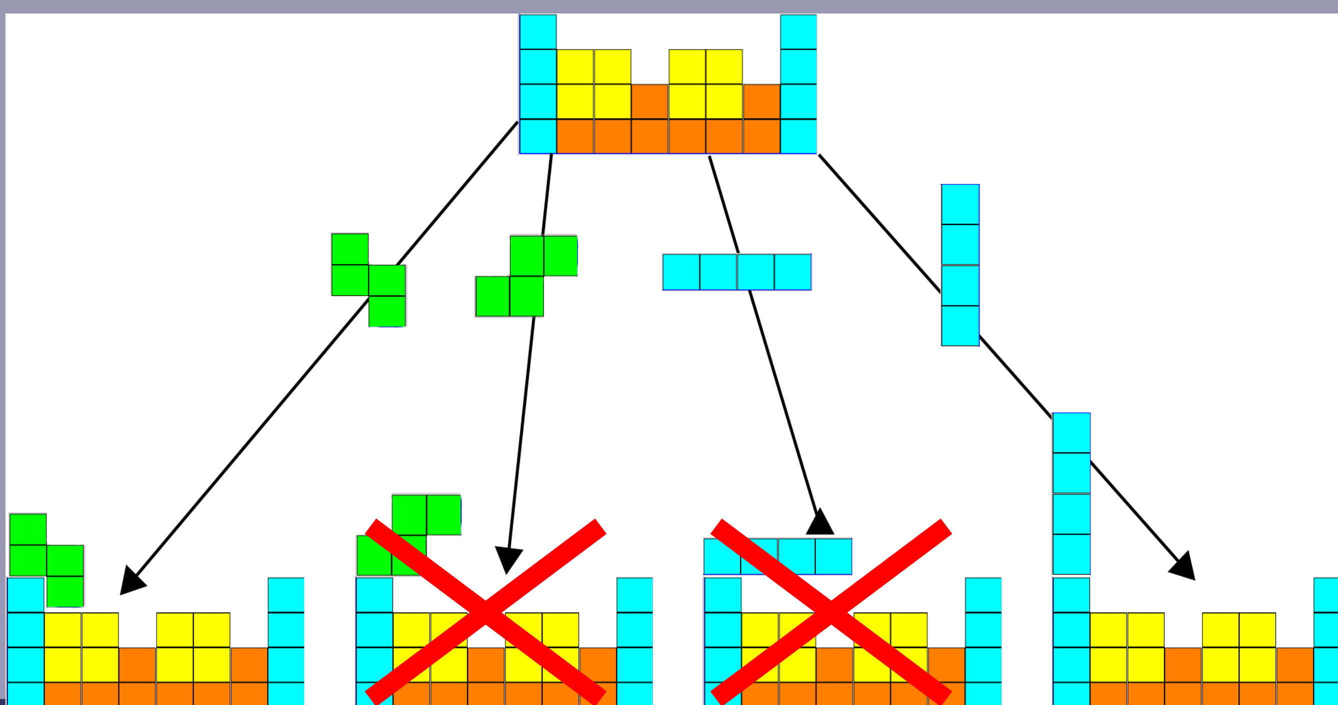
Building the Tree – 2

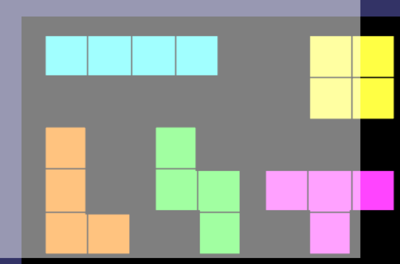
```
void addActionsToSet(...) {
    . . .
    addSActionsToSet(board, location, actions);
    addIActionsToSet(board, location, actions);
    . . .
```



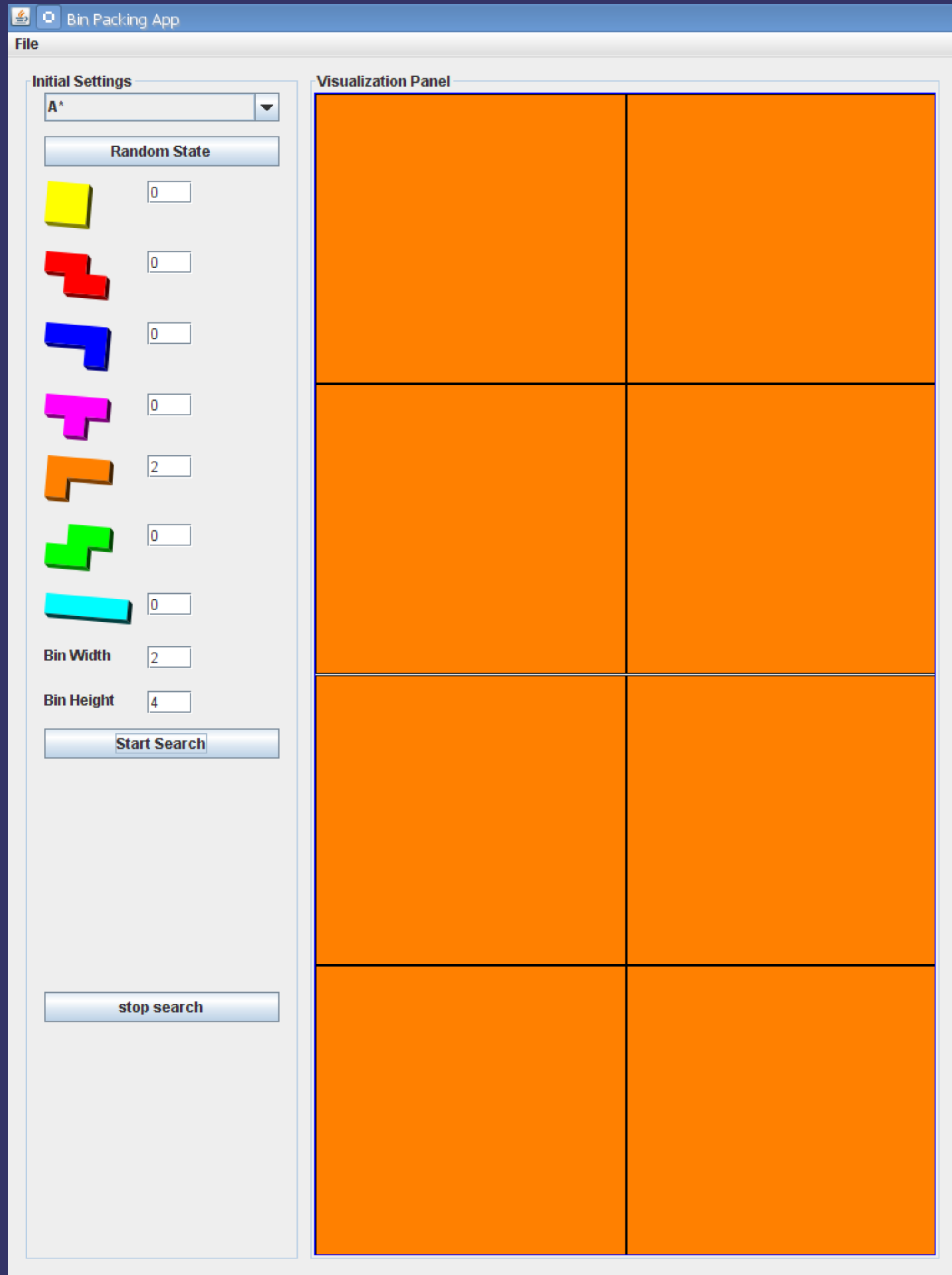
Building the Tree – 3

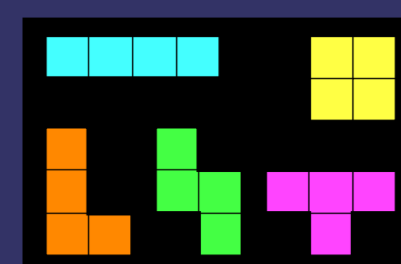
```
addSActionsToSet(TetrisBoard board, XYLocation loc, Set<Action> actions)
{
  TetrisPiece piece = TetrisPiece.S_PIECE;
  if (board.willFit(piece, loc)
      && !BadMoveTest.isBadMove(board, piece, loc)) {
        actions.add(TetrisPieceAction.PLACE_S, loc);
  }
  // try rotate and place S piece
  piece.rotateClockwise();
  if (board.willFit(piece, loc)
      && !BadMoveTest.isBadMove(board, piece, loc)) {
        actions.add(TetrisPieceAction.ROTATE_AND_PLACE_S, loc);
  }
}
```



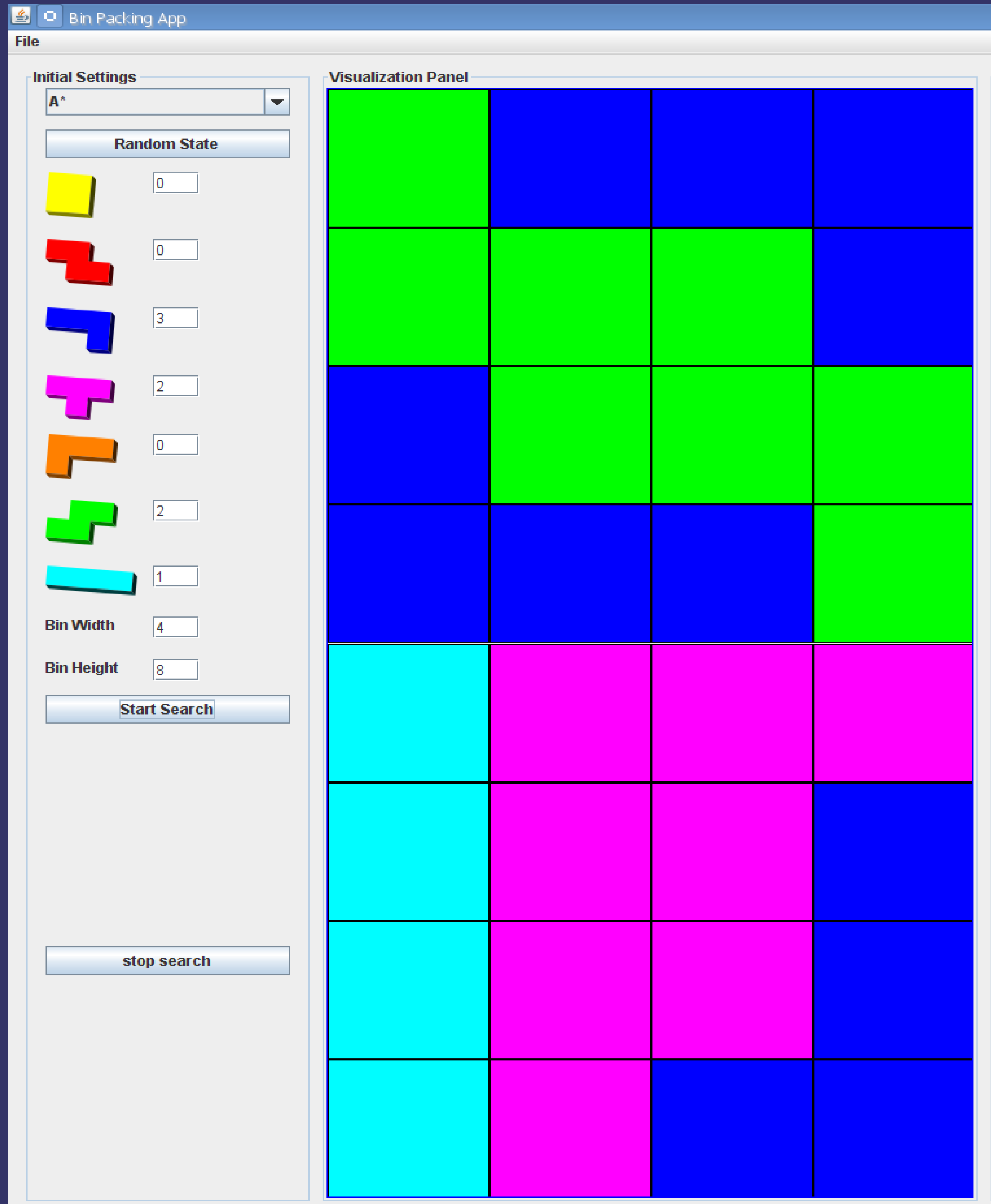


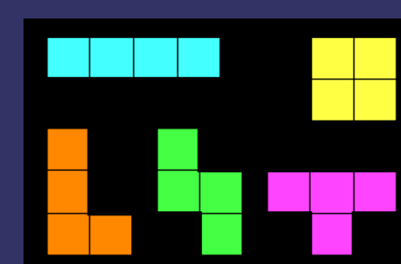
Screen Shot #1 – 2x4



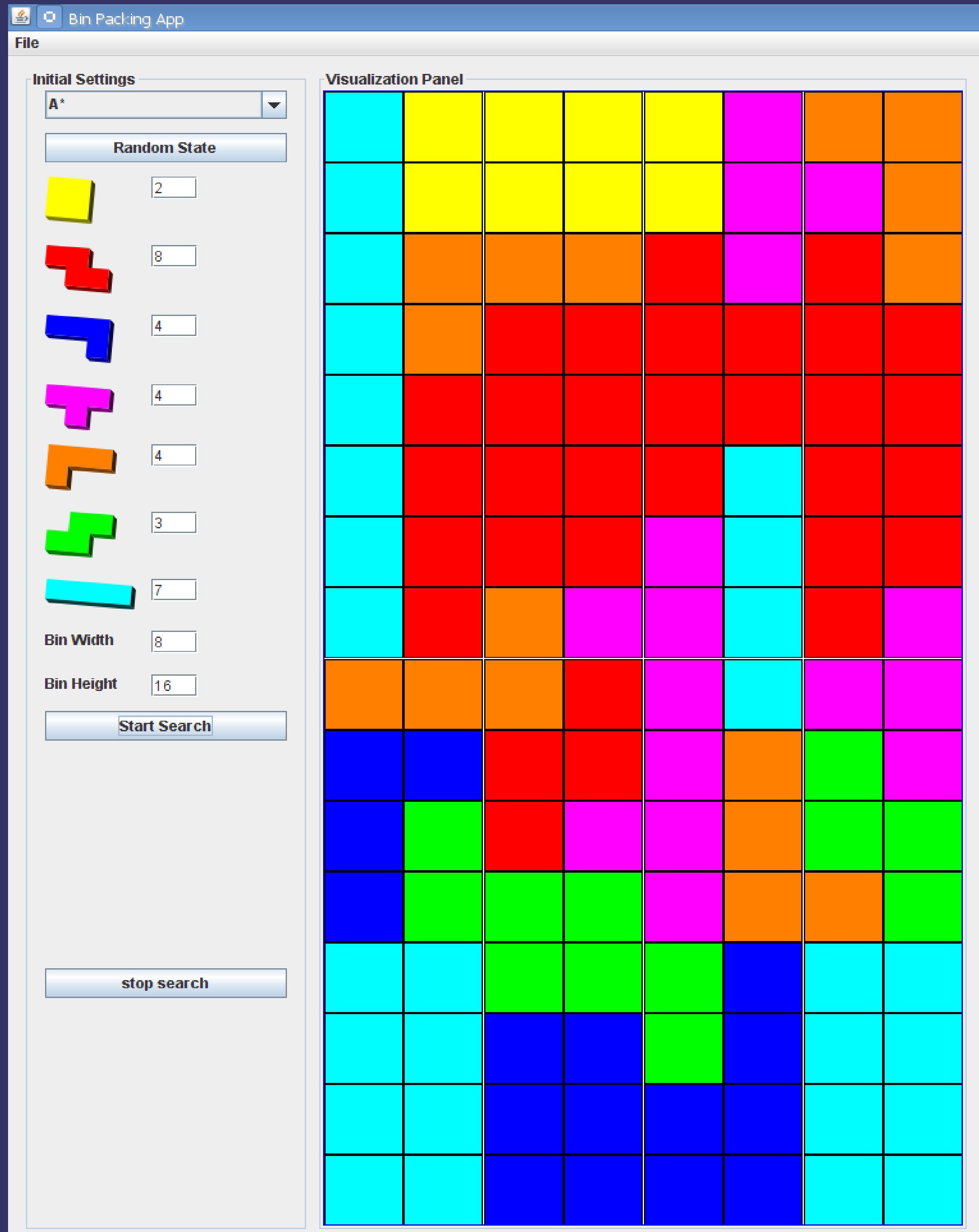


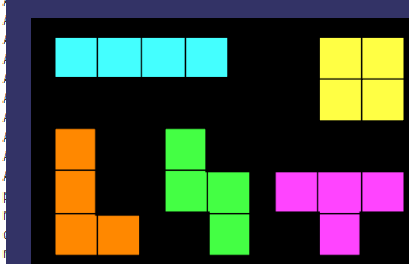
Screen Shot #1 – 4x8



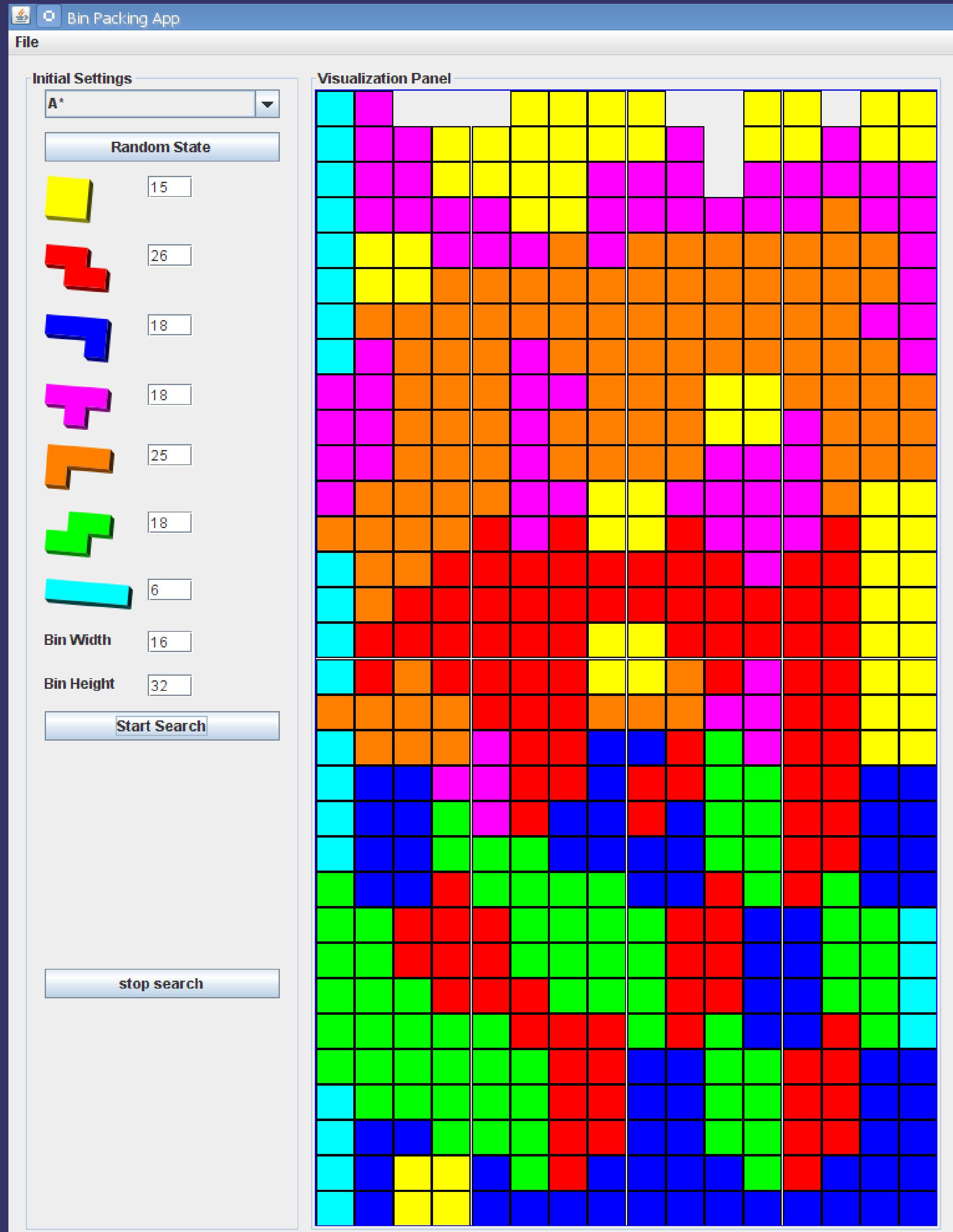


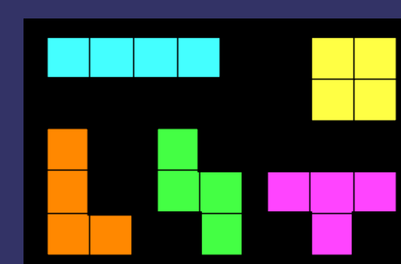
Screen Shot #1 – 8x16



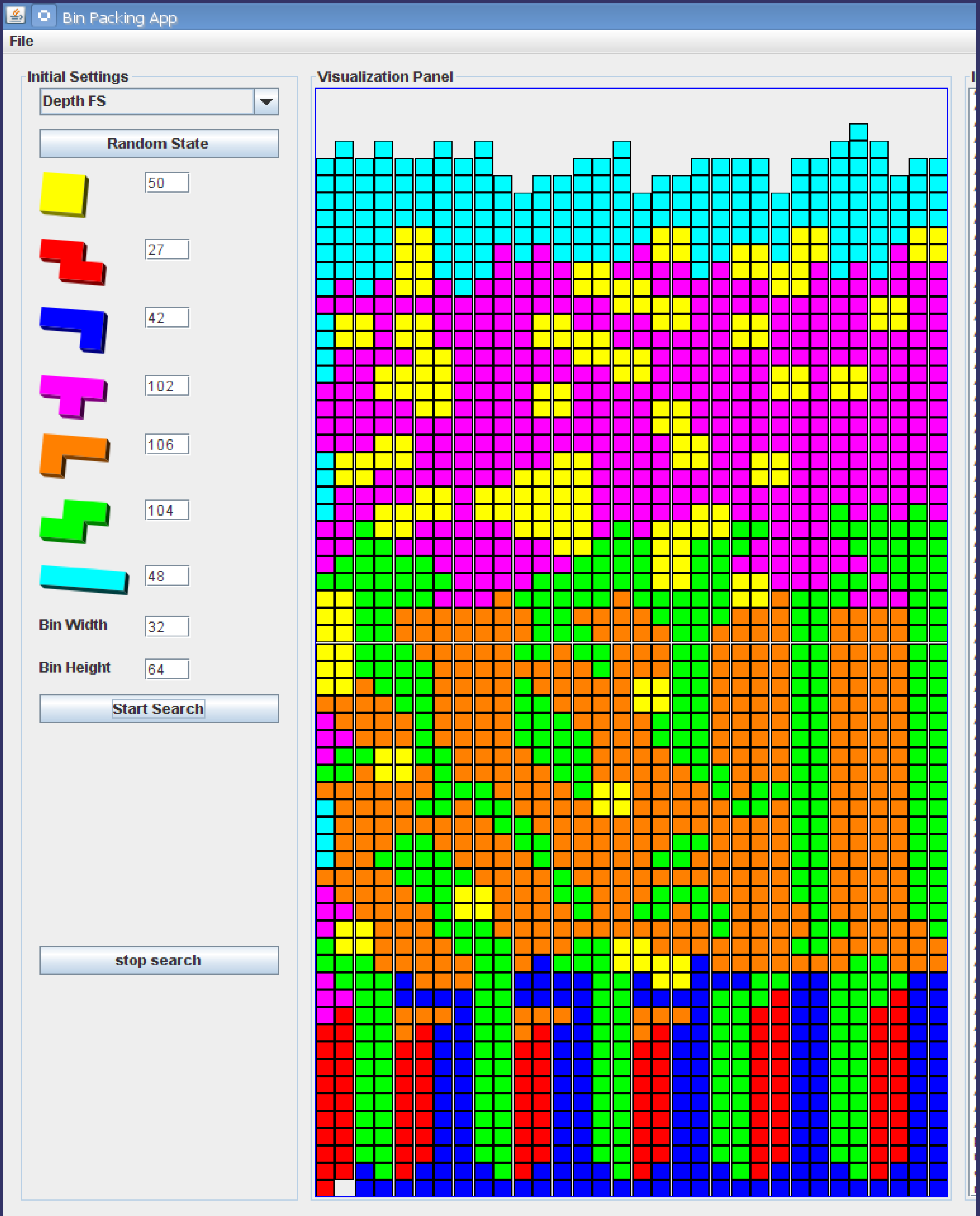


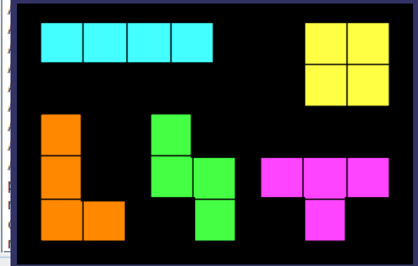
Screen Shot #1 – 16x32



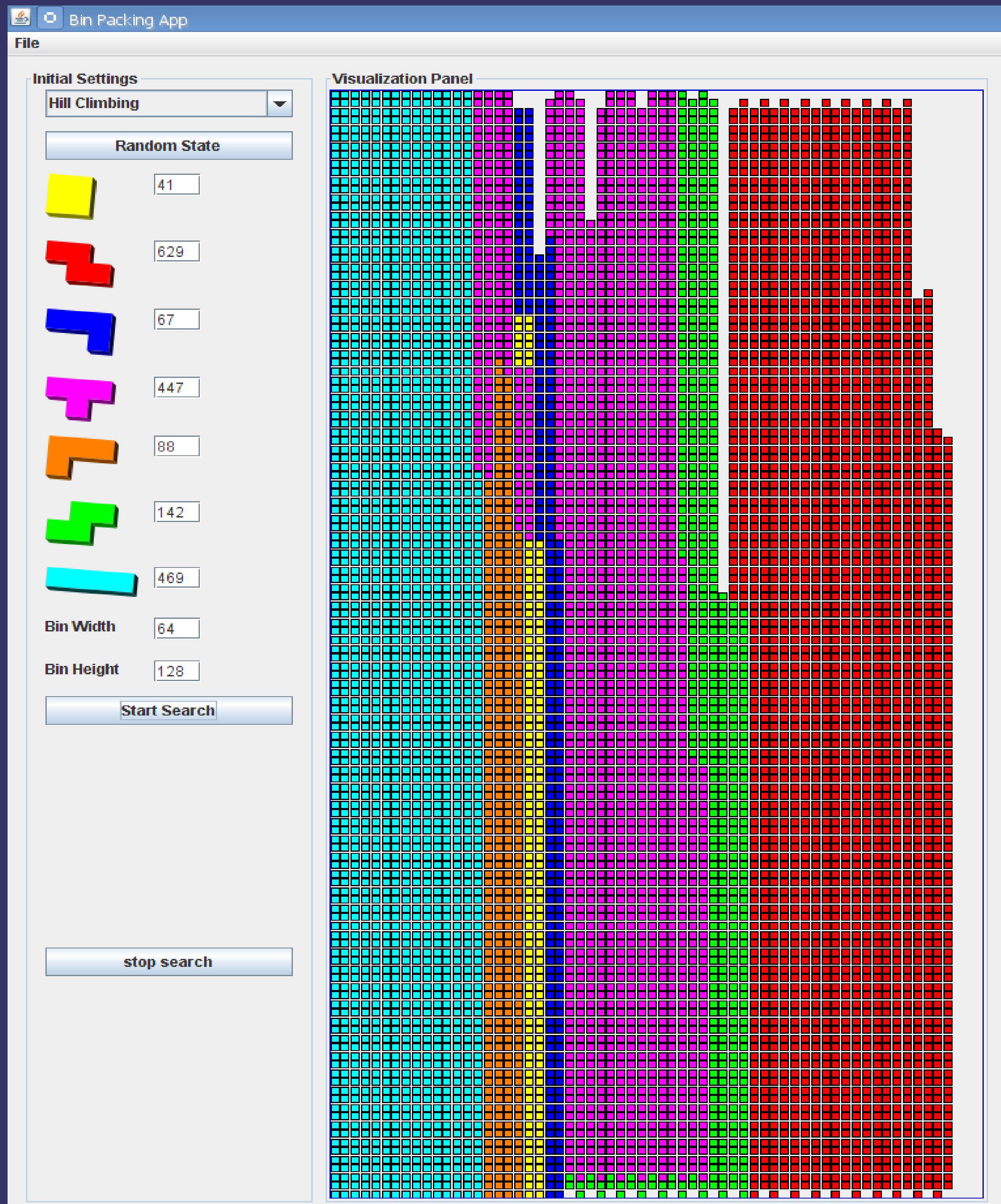


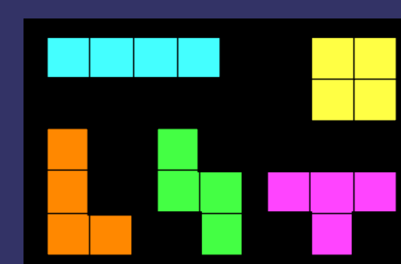
Screen Shot #1 – 32x64



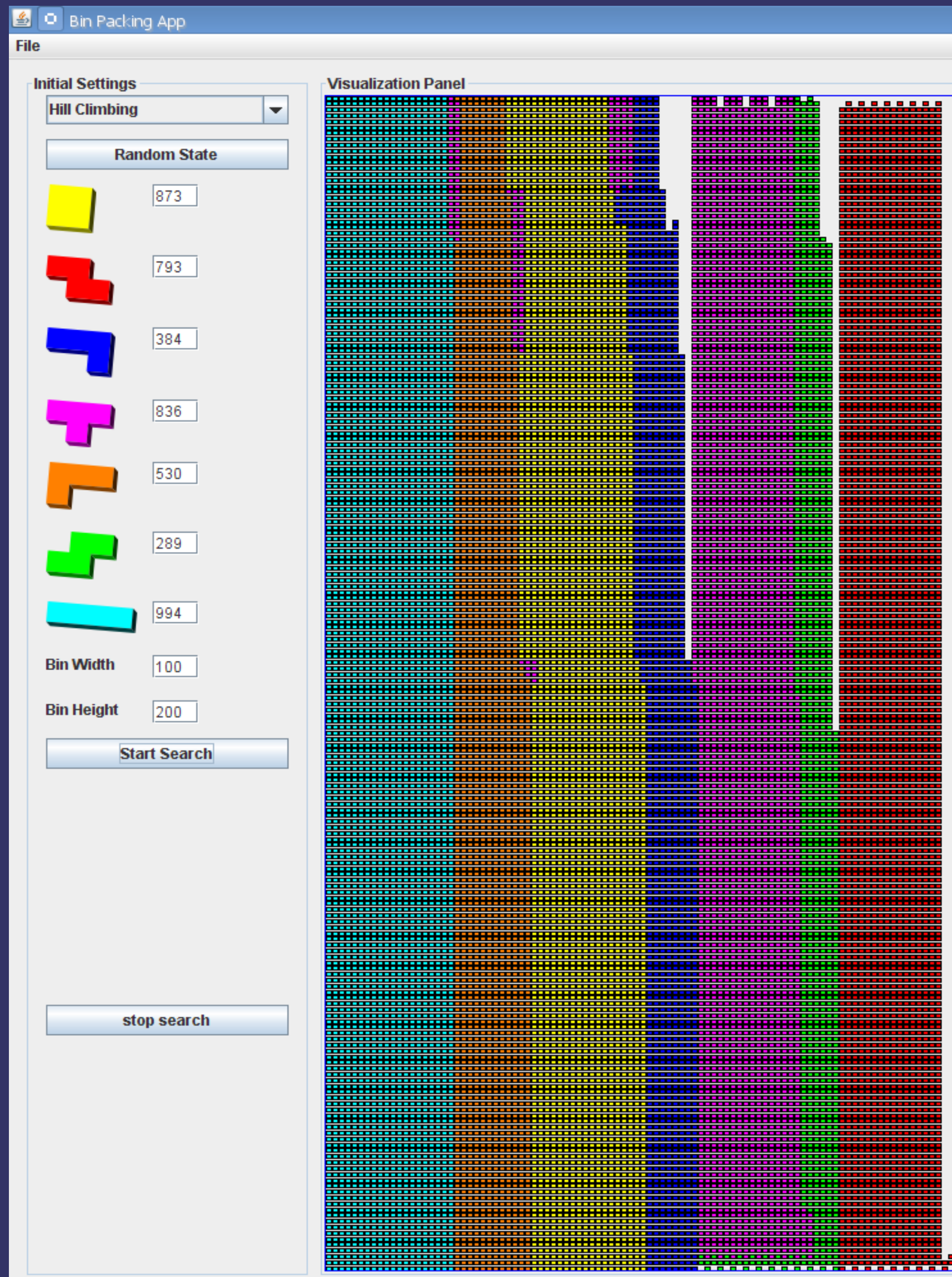


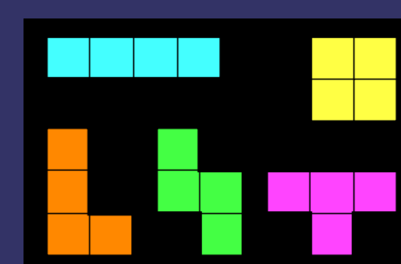
Screen Shot #1 – 64x128

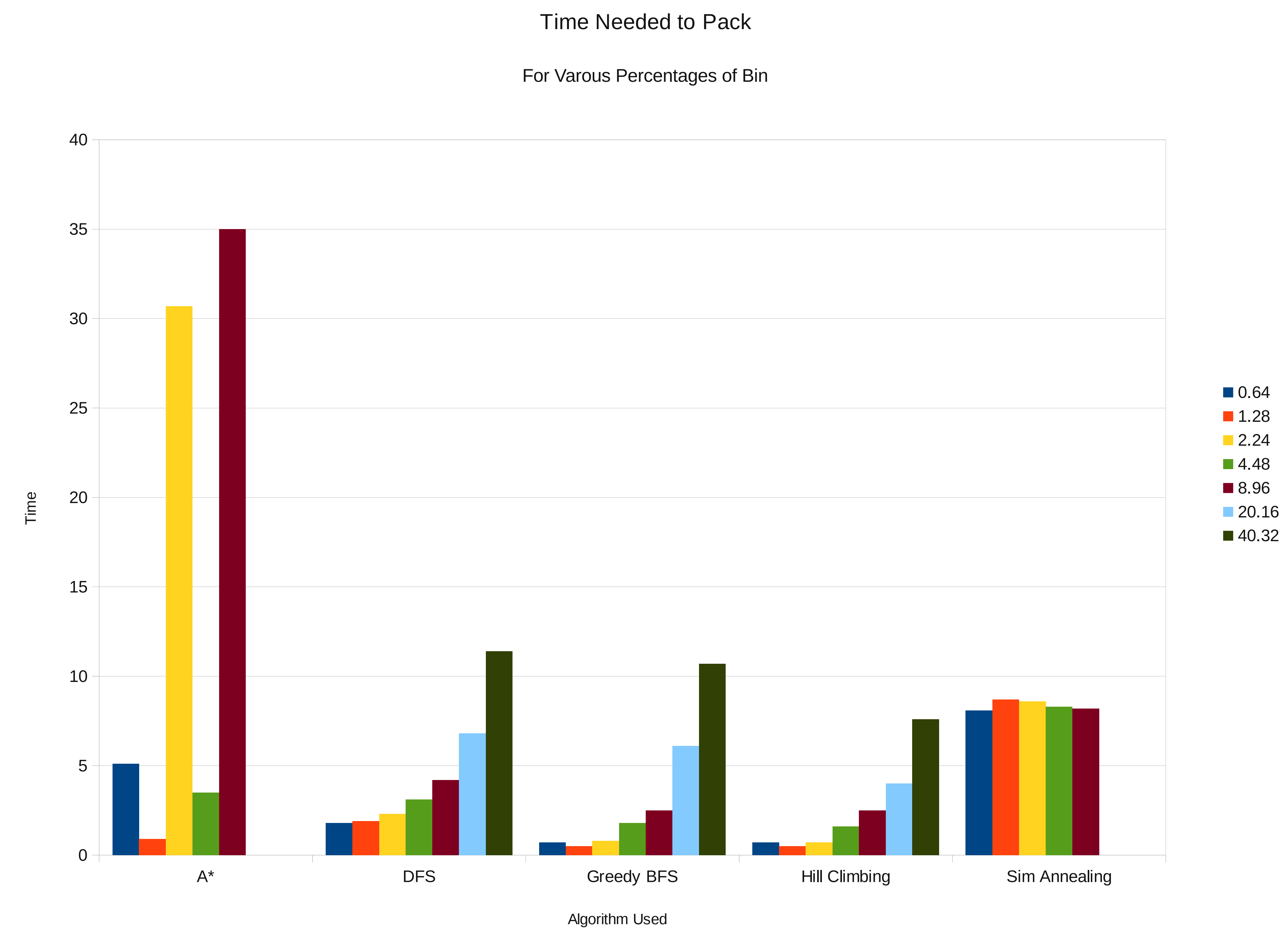




Screen Shot #1 – 100x200






Time Needed to Pack
For Varous Percentages of Bin
Time
40
35
30
25
20
15
10
5
0
A*
DFS
Greedy BFS
Hill Climbing
Sim Annealing
Algorithm Used
0.64
1.28
2.24
4.48
8.96
20.16
40.32

Question Time