Discrete Optimization

MA2827

Fondements de l'optimisation discrète

Dynamic programming (Part 2)

https://project.inria.fr/2015ma2827/

Material based on the lectures of Erik Demaine at MIT and Pascal Van Hentenryck at Coursera

Outline

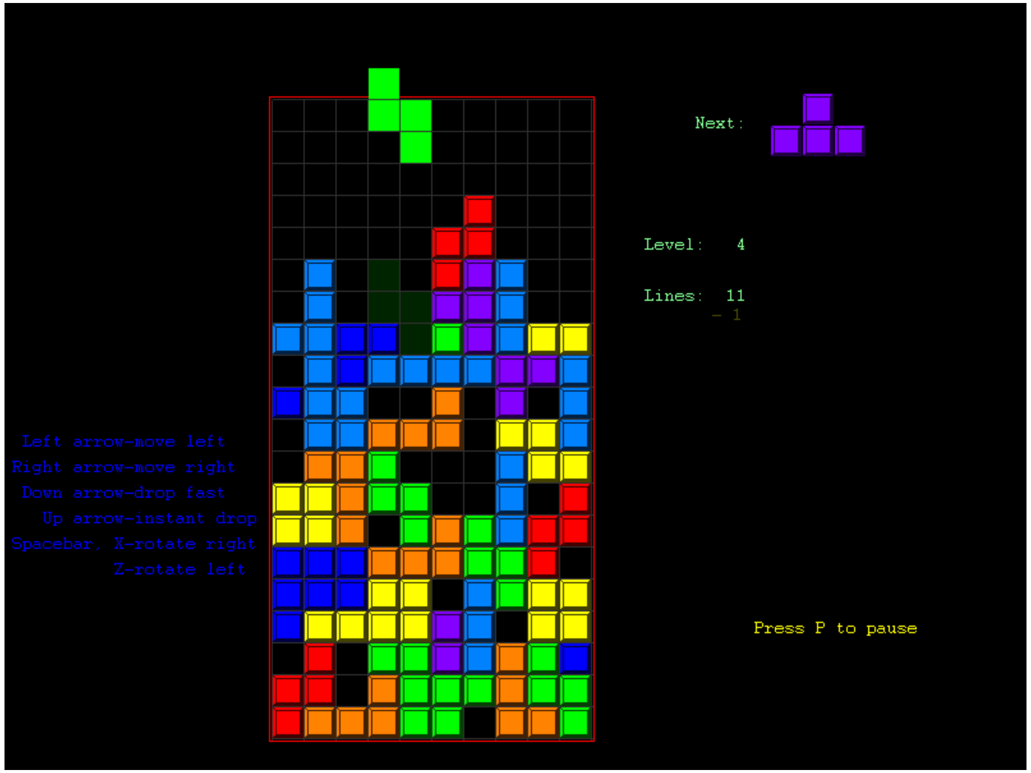
- Dynamic programming
  - Guitar fingering
- More dynamic programming
  - Tetris
  - Blackjack

Quiz (as homework)

- Write the DP for perfect-information blackjack
- Derive the number of subproblems for the tetris problem

Tetris

Task: win in the game of Tetris!



Next:

Level: 4

Lines: 11
– 1

Left arrow-move left
Right arrow-move right
Down arrow-drop fast
Up arrow-instant drop
Spacebar, X-rotate right
Z-rotate left

Press P to pause

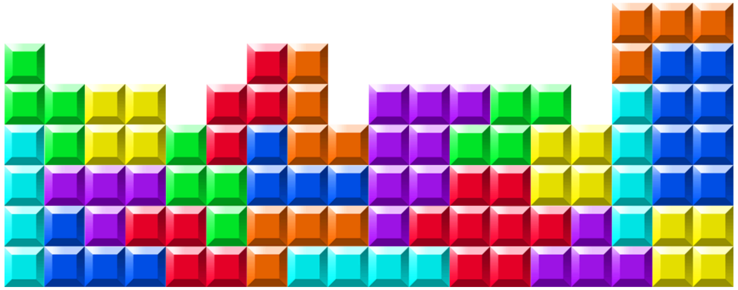
Tetris

Task: win in the game of Tetris!

- Input: a sequence of n Tetris pieces and an empty board of small width w
- Choose orientation and position for each piece
- Must drop piece till it hits something
- Full rows do not clear
- Goal: survive i.e., stay within height h

Tetris

Task: stay within height h



Subproblem:
survival? in suffix [i :]
given a particular column profile
#subproblems = O(n $(h+1)^w$)

Guesses:
where to drop piece i?
#choices = O(w)

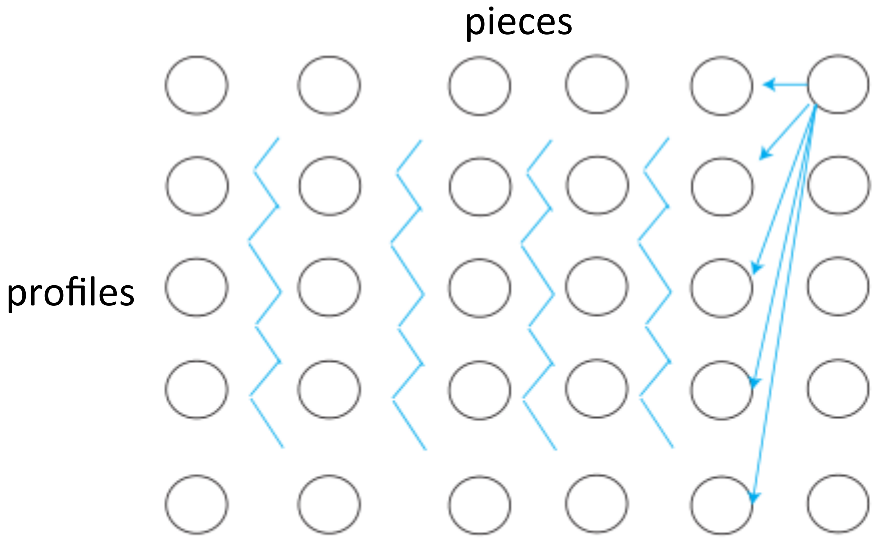
Recurrence:
DP[i, p] = max { DP[i + 1, q] | q is a valid move from p }
Base-case: DP[n+1, p] = true for all profiles p
time/subproblem = O(w)

Tetris

Task: stay within height h

Topological order:

```
for i = n – 1, n – 2, …, 0:
    for p = 0, …, $(h+1)^w – 1$:
```



total time $O(\ n\ w\ (h+1)^w\ )$

Final problem:

DP[0, empty]

Blackjack

Task: beat the blackjack (twenty-one)!

Blackjack

Task: beat the blackjack!

Rules of Blackjack (simplified):

- The player and the dealer are initially given 2 cards each
- Each card gives points:
    - Cards 2-10 are valued at the face value of the card
    - Face cards (King, Queen, Jack) are valued at 10
    - The Ace card can be valued either at 11 or 1
- The goal of the player is to get more points than the dealer, but less than 21, if more than 21 than he looses (busts)
- Player can take any number of cards (hits)
- After that the dealer hits deterministically: until ≥ 17 points

Perfect-information Blackjack

Task: beat the blackjack with a marked deck!

- Input: a deck of cards $c_0, \ldots, c_{n-1}$
- Player vs. dealer one-on-one
- Goal: maximize winning for a fixed bet \$1
- Might benefit from loosing to get a better deck

Perfect-information Blackjack

Task: beat the blackjack with a marked deck!

Subproblem:
BJ(i) = best play of $c_i, \ldots, c_{n-1}$
#subproblems = O(n)

Guesses:
how many times player hits?
#choices ≤ n

Topological order:

Final problem:

Recurrence:
BJ(i) = max{ outcome ∈ {-1, 0, 1} + BJ(i + 4 + #hits + #dealer hits)
| for #hits = 0, …, n if valid play }

Perfect-information Blackjack

Detailed recursion:

```
def BJ(i):
    if $n - i < 4$: return 0      (not enough cards)
    outcome = [ ]
    for $p = 2, \ldots, n - i - 2$:    (# cards taken)
        player = $c_i + c_{i+2} + c_{i+4} + \ldots + c_{i+p+2}$
        if player > 21:          (bust)
            outcome.append(  -1  + BJ(i+p+2) )
            break
        for $d = 2, \ldots, n - i - p - 1$
            dealer = $c_{i+1} + c_{i+3} + c_{i+p+2} + \ldots + c_{i+p+d}$
            if dealer $\geq$ 17: break
        if dealer > 21: dealer = 0  (bust)
        outcome.append(  cmp(player, dealer) + BJ(i + p + d) )
    return max( outcome )
```

Perfect-information Blackjack

Task: beat the blackjack with a marked deck!

Subproblem:
BJ(i) = best play of $c_i, \ldots, c_{n-1}$
#subproblems = O(n)

Guesses:
how many times player hits?
#choices ≤ n

Recurrence:
BJ(i) = max{ outcome ∈ {-1, 0, 1} + BJ(i + 4 + #hits + #dealer hits)
| for #hits = 0, …, n if valid play }
time/subproblem = O(n^2)

Topological order:
for i = n-1, …, 0:

total time O(n^3)

Final problem:
BJ[0]