CSCI203

Week 11 – Lecture B

Blackjack

Blackjack

- Blackjack, also known as pontoon, *vingt et un* or 21 is a game played with one or more decks of cards between two or more players; one of whom, the dealer, plays in a purely deterministic manner.
- The objective is to get a hand with a closer value to the maximum score of 21 than the dealer.
- A hand over 21 always loses.
- All face cards have a value of 10.
- Aces may be counted as either 1 or 11 at the player’s discretion.
- Cards 2 through 10 have their face value.

Play: 1

- The best hand is blackjack: Ace and Ten (10, J, Q, K).
  - If both player and dealer have blackjack the game is drawn.
- Other than this, the higher scoring hand wins.
- Game starts with player and dealer each receiving two cards:
  - One of the dealer’s cards, the “hole card”, is dealt face down;
  - Both of the player’s cards are dealt face up.
- Player plays first:
  - Hit: take another card;
  - Stand: do not take another card.

Play: 2

- If the player “busts”, scores more than 21, the dealer has won.
- Otherwise the dealer plays.
- He flips the hole card:
  - If it reveals a blackjack or a total greater than the player, he wins;
  - If it scores less than 17 he must hit;
  - If it scores 17 or more he must stand.
- Rules on “soft” 17 (Ace, 6) vary from casino to casino but are always fixed on either hit or stand.

The Odds:

- In a fair game of blackjack, assuming the player always makes the optimal decision based on the cards revealed, the house has a slight advantage.
- Typically, depending on specific house rules, the return is around 1%:
  - For every $100 bet you can expect to win back around $99.
- Players can significantly improve their odds by a technique known as card counting:
  - Keeping track of the cards that have been dealt so far and adjusting play as the game progresses.
- Casinos typically blacklist players who are suspected of card counting!
- Or worse!

Blackjack and Dynamic Programming

- In this lecture we are going to cheat at blackjack to maximize the player's possible profit.
- We are going to solve the game of blackjack with perfect knowledge.
- What is the best play on each hand if we know in advance the entire sequence of cards in the deck?
- Needless to say, this may prove a little difficult in a real casino. ☺

Dynamic Programming VI
Cheating at Blackjack

DP VI: Perfect Knowledge Blackjack

- Deck: $C_0, C_1, \ldots, C_{n-1}$ is known before play starts.
- Single player.
  - Solution is not known for more than one player.
- \$1 bet per hand.
- Payoff is always \$1 (No bonus for blackjack).
- No special rules (double, split, insurance, etc.)
- Let us now proceed to the DP analysis of the problem.

1, 2. Identify the Sub-Problems and the Guess

- In this case the sub-problem is where to start the next hand.
- Number of subproblems:
  - n.
- The guess involves the number of times I should hit in the current hand:
- This is a number between 0 and the number of cards which causes me to bust.
  - Number of guesses $\leq n$.

3. Relate Sub-Problems via Recursion

- Our sub-problem involves finding the optimal play for the current hand such that the profit from the remaining deck is optimized.
- If the current deck starts at the i^{th} card we need to find BJ(i).
- **Procedure BJ(i)**

```
Procedure BJ(i)
    if i ≥ n return 0
    profit = -∞
    for h=0 to n
        profit = max(profit, outcome(h)+BJ(j))
    rof
    return profit
end BJ
```

Notes

- Note: the relationship between h, the number of times I hit, and j, the identity of the card after the next deal, is somewhat complicated as it depends on the dealer's play but, as this is deterministic, it can be readily worked out.
- In general, $j = i+h+4+d$, where d is the (deterministic) number of cards taken by the dealer.
  - The number of cards the dealer takes may depend on the house rule regarding a score of 17.
- Note: the function **`outcome(k)`** $\in \{-1, 0, 1\}$ and is easily determined because the dealer has no freedom of choice.

4, 5 Find the Topological Order, Solving the Problem

- In this case, to compute **`BJ(i)`**, we need to evaluate **`BJ(j)`** where $j > i$.
- Once again, this means that we need to evaluate the sub-problems in reverse order, starting at **`BJ(n)`** and working back to **`BJ(0)`**.
- The topological order is, again, simply n, $n-1$, …, 2, 1, 0.
- Our total running time:
  - # sub-problems $\times$ time per sub-problem;
  - $n \times O(n)$;
  - $O(n^2)$.
- The final solution is **`BJ(0)`**.