D: Dicey Decision

Problem author: Freek Henstra

Problem: Given a *Snakes and Ladders*-type board with n spaces, m shortcuts, turn limit t, and four six-sided dice to choose from. If you can pick your die first and you start playing last, what is your maximum possible probability of winning?

D: Dicey Decision

Problem author: Freek Henstra

Problem: Given a *Snakes and Ladders*-type board with n spaces, m shortcuts, turn limit t, and four six-sided dice to choose from. If you can pick your die first and you start playing last, what is your maximum possible probability of winning?

Naive solution: Dynamic programming on the state space. There are $\mathcal{O}(tn^4)$ possible states, which is too many. (State is number of turns to go and each player's position.)

D: Dicey Decision

Problem author: Freek Henstra

Problem: Given a *Snakes and Ladders*-type board with n spaces, m shortcuts, turn limit t, and four six-sided dice to choose from. If you can pick your die first and you start playing last, what is your maximum possible probability of winning?

Naive solution: Dynamic programming on the state space. There are $\mathcal{O}(tn^4)$ possible states, which is too many. (State is number of turns to go and each player's position.)

Observation: The players don't interact with one another, so we can consider each player separately.

D: Dicey Decision

Problem author: Freek Henstra

Problem: Given a *Snakes and Ladders*-type board with n spaces, m shortcuts, turn limit t, and four six-sided dice to choose from. If you can pick your die first and you start playing last, what is your maximum possible probability of winning?

Naive solution: Dynamic programming on the state space. There are $\mathcal{O}(tn^4)$ possible states, which is too many. (State is number of turns to go and each player's position.)

Observation: The players don't interact with one another, so we can consider each player separately.

Solution part 1: Dynamic programming for each die separately. For each die, calculate each value of $P(i, j)$: The probability of landing on space i after j turns.

D: Dicey Decision

Problem author: Freek Henstra

Problem: Given a *Snakes and Ladders*-type board with n spaces, m shortcuts, turn limit t, and four six-sided dice to choose from. If you can pick your die first and you start playing last, what is your maximum possible probability of winning?

Naive solution: Dynamic programming on the state space. There are $\mathcal{O}(tn^4)$ possible states, which is too many. (State is number of turns to go and each player's position.)

Observation: The players don't interact with one another, so we can consider each player separately.

Solution part 1: Dynamic programming for each die separately. For each die, calculate each value of $P(i, j)$: The probability of landing on space i after j turns.

Recursion: For each i, j with j from small to large, starting on space i, determine for each possible die roll the landing space k, and add $\frac{1}{6}P(i, j)$ to $P(k, j + 1)$.

D: Dicey Decision

Problem author: Freek Henstra

Problem: Given a *Snakes and Ladders*-type board with n spaces, m shortcuts, turn limit t, and four six-sided dice to choose from. If you can pick your die first and you start playing last, what is your maximum possible probability of winning?

Naive solution: Dynamic programming on the state space. There are $\mathcal{O}(tn^4)$ possible states, which is too many. (State is number of turns to go and each player's position.)

Observation: The players don't interact with one another, so we can consider each player separately.

Solution part 1: Dynamic programming for each die separately. For each die, calculate each value of $P(i, j)$: The probability of landing on space i after j turns.

Recursion: For each i, j with j from small to large, starting on space i, determine for each possible die roll the landing space k, and add $\frac{1}{6}P(i, j)$ to $P(k, j + 1)$.

Solution part 2: For each die independently, calculate the probability of winning (with tie-break rules). For each die, determine after each turn the probability of finishing on that turn with no other player having finished yet, and after t turns, determine for each space the probability of landing there ahead of all other players.

D: Dicey Decision

Problem author: Freek Henstra

Problem: Given a *Snakes and Ladders*-type board with n spaces, m shortcuts, turn limit t, and four six-sided dice to choose from. If you can pick your die first and you start playing last, what is your maximum possible probability of winning?

Naive solution: Dynamic programming on the state space. There are $\mathcal{O}(tn^4)$ possible states, which is too many. (State is number of turns to go and each player's position.)

Observation: The players don't interact with one another, so we can consider each player separately.

Solution part 1: Dynamic programming for each die separately. For each die, calculate each value of $P(i, j)$: The probability of landing on space i after j turns.

Recursion: For each i, j with j from small to large, starting on space i, determine for each possible die roll the landing space k, and add $\frac{1}{6}P(i, j)$ to $P(k, j + 1)$.

Solution part 2: For each die independently, calculate the probability of winning (with tie-break rules). For each die, determine after each turn the probability of finishing on that turn with no other player having finished yet, and after t turns, determine for each space the probability of landing there ahead of all other players.

Running time: $\mathcal{O}(tn)$ (times #players and #sides-per-die, which are constant).

D: Dicey Decision

Problem author: Freek Henstra

Problem: Given a *Snakes and Ladders*-type board with n spaces, m shortcuts, turn limit t, and four six-sided dice to choose from. If you can pick your die first and you start playing last, what is your maximum possible probability of winning?

Naive solution: Dynamic programming on the state space. There are $\mathcal{O}(tn^4)$ possible states, which is too many. (State is number of turns to go and each player's position.)

Observation: The players don't interact with one another, so we can consider each player separately.

Solution part 1: Dynamic programming for each die separately. For each die, calculate each value of $P(i, j)$: The probability of landing on space i after j turns.

Recursion: For each i, j with j from small to large, starting on space i, determine for each possible die roll the landing space k, and add $\frac{1}{6}P(i, j)$ to $P(k, j + 1)$.

Solution part 2: For each die independently, calculate the probability of winning (with tie-break rules). For each die, determine after each turn the probability of finishing on that turn with no other player having finished yet, and after t turns, determine for each space the probability of landing there ahead of all other players.

Running time: $\mathcal{O}(tn)$ (times #players and #sides-per-die, which are constant).

Statistics: 49 submissions, 3 accepted, 38 unknown