MECH 6326 - Optimal Control and Dynamic Programming
Final Project Working Doc

Alyssa Vellucci and Jonas Wagner

April 16, 2023

Chapter 1

Simple System Model

1.1 System Definition

1.1.1 Assumptions:

- Movement: Single movement per turn
  - Deterministic
  - 1 square movement
  - move then action
- Actions: Single action per time step
  - Melee (hit check)(d6) - Short range
  - Ranged (hit check)(d8) - Longer range
  - Health Potion (d4 + 1)
  - Nothing
- Characters
  - 1 PC and 1 Monster
  - Identical Specs/modifiers
  - Same weapon (+2)
- Monster
  - Monster move in standard pattern
  - Monster cannot heal
- Infinite Time Horizon
- Infinite Battlefield and no Obstacles

1.1.2 Environment Definition

States

Let each character be associated with position and HP states. For position, let

$$x_{pc,p}, x_{mn,p} \in \mathcal{X}_p \subseteq \mathbb{Z}^2$$

describe the position on an infinite 2-d grid. For HP, let

$$x_{pc,hp}, x_{mn,hp} \in \mathcal{X}_{hp} \subseteq \mathbb{Z}_+ = \{0, 1, 2, \dots\}$$

describe the HP for each character.

Inputs

The inputs to the system consist of movement and actions impacting the position and hp states respectively. For movement, a deterministic input of

$$\begin{aligned} u_{pc,m}, u_{mn,m} \in \mathcal{U}_m &= \{\text{N, E, S, W, NE, NW, SE, SW}\} \\ &= \{(-1,0),(+1,0),(0,-1),(0,+1),(-1,-1),(-1,+1),(+1,-1),(+1,+1)\} \end{aligned} \quad (1.1)$$

For actions, all the actions (except nothing) each are stochastic and can be represented as Markov chains or as a combination of input and noise term, $u_{pc,a}, u_{mn,a} \in \mathcal{U}_a = \{\text{Melee, Ranged, Heal, Nothing}\}$.

For Melee and Ranged attacks, the character acts upon another character's HP where the impact on HP is as follows:

1. Ensure in range for either melee or ranged attack - otherwise can't attack.
2. "Roll" for success/fail - if fail then self-loop on opponent HP
3. "Roll" for effectiveness - opponent HP decreased by Weapon/self Modifiers (2) + d6/d8

The PC is allowed to use a health potion which has a stochastic effect upon the player's health:

1. Ensure potion is available - otherwise can't heal
2. "Roll" for effectiveness - player's HP increased by health modifier (1) + d4

Noise

When written in some forms the stochastic aspects of the system can be described as a noise signal. This can be modeled as either a single HP update amount with dependent distribution on action or as a non-addative noise signal where the update is dependent as described.

For a complete picture, let the action noise be described as

$$w_{pc,a}, w_{mn,a} \in \mathcal{W}_a \subseteq \{\text{"success"} = 1, \text{"failure"} = 0\} \times \mathbb{Z}_+$$

where the distributions of each are dependent on the input action $u_{pc,a}, u_{mn,a}$ and associated attack modifiers.

Alternativly for the simplistic case this can be done separately, where {"success", "failure"} is dependent on the PC and monster's stats as well as their respective dice rolls, thus

$$w_{pc,sf}, w_{mn,sf} \in \mathcal{W}_{sf} \subseteq \{0,1\}$$

and where $w_{pc,sf}$ and $w_{mn,sf}$ are directly calculated from PC and monster modifiers and a d20 role. The damage or heal amount is then also stochastic dependent on player stats and also dependent on modifiers and a dice roll (d4/d6/d8).

The dice roll noise is defined for dice {d2, d4, d6, d8, d10, d20, d100} as

$$w_{pc,dn}, w_{mn,dn} \in \mathcal{W}_{dn} = \{1, 2, \ldots, n\}$$

with $n = 2, 4, 6, 8, 10, 20, 100$ respectively and each outcome is equally likely. The PDF PMF? can be seen in shorthand with $P(w_{i,dn} = [1 \cdots n]^T) = \frac{1}{n}[1 \cdots 1]^T$

1.1.3 Problem Statement

For the simplistic case, let states at time-step k, be

$$x_k = \begin{bmatrix} x_{pc,p} \\ x_{mn,p} \\ x_{pc,hp} \\ x_{mn,hp} \\ x_{pc,potion} \end{bmatrix} \in \mathcal{X} = \mathcal{X}_p^2 \times \mathcal{X}_{hp}^2 \subseteq \mathbb{Z}^4 \times \mathbb{Z}_+^2 \times \mathbb{Z}_+$$

where states and sets are defined as before and $x_{pc,potion}$ is the number of potions available to the PC.

Let the inputs to the system be only the players inputs

$$u_k = \begin{bmatrix} u_{pc,m} \\ u_{pc,a} \end{bmatrix} \in \mathcal{U} = \mathcal{U}_p \times \mathcal{U}_a$$

The monster's inputs to the system will be incorporated as a deterministic input that and stochastic input that are closed-loop within the system and treated as part of the nonlinear aspects of the update function/Markov chains.

Let the stochastic input signal w_k for each time-step be defined as a collection of all the dice rolls

$$w_k = \begin{bmatrix} w_{pc} \\ w_{mn} \end{bmatrix}, \quad w_i = \begin{bmatrix} w_{i,4} \\ w_{i,6} \\ w_{i,8} \\ w_{i,20} \end{bmatrix} \forall_{i=pc,mn}$$

and the associated success/fail noise $\forall_{i=pc,mn}$ be derived as

$$w_{i,sf} = \begin{cases} 0 & (5 + w_{i,d20} < 15 \text{ OR } w_{i,d20} = 1) \text{ AND } w_{i,d20} \neq 1 \\ 1 & (5 + w_{i,d20} \geq 15 \text{ OR } w_{i,d20} = 20) \text{ AND } w_{i,d20} \neq 1 \end{cases}$$

This results in probabilities of $P(w_{i,sf} = 0) = (P(w_{i,d20} = 1) + P(5 + w_{i,d20} < 15)) - (P(w_{i,d20} = 20)) = 1/20 + 9/20 - 1/20 = 9/20$ and $P(w_{i,sf} = 1) = (P(w_{i,d20} = 20) + P(5 + w_{i,d20} \geq 15)) - (P(w_{i,d20} = 1)) = 1/20 + 11/20 - 1/20 = 11/20$. In shorthand, $P(w_{i,sf} = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right]) = \left[\begin{smallmatrix} 0.45 \\ 0.55 \end{smallmatrix}\right]$.

define modifiers as a constant...

The evolution of the very simple system can be described as Markov chains or by a nonlinear update function:

$$x_{k+1} = f(x_k, u_k, w_k) = \begin{bmatrix} x_{pc,p} + f_{pc,m}(u_k) \\ x_{mn,p} + f_{mn,m}(x_k) \\ \begin{bmatrix} x_{pc,hp} \\ x_{mn,hp} \\ x_{pc,potion} \end{bmatrix} + f_{pc,a}(x_k, u_k, w_k) + f_{mn,a}(x_k, w_k) \end{bmatrix}$$

where the associated functions update states as follows:

- The players deterministic movement input: $f_{pc,m}(u_k) = u_{pc,m}$
- The monsters state-dependent movement:

  $$f_{mn,m}(x_k) = \text{direction}(x_{pc,p} - x_{mn,p})$$

  where direction() is calculated as the closest cardinal direction that heads towards the player. (In matlab we do: `round(k_mn_speed * normalize(x_pc_p-x_mn_p))`)
- The players action:

  $$f_{pc,a} = \begin{cases} \begin{bmatrix} 0 & -w_{pc,sf}(2 + w_{pc,d6}) & 0 \end{bmatrix}^T & u_{pc,a} = \text{Melee AND } \|x_{pc,p} - x_{mn,p}\|_1 \leq \text{Melee Range} \\ \begin{bmatrix} 0 & -w_{pc,sf}(2 + w_{pc,d8}) & 0 \end{bmatrix}^T & u_{pc,a} = \text{Ranged AND } \|x_{pc,p} - x_{mn,p}\|_1 \leq \text{Ranged Range} \\ \begin{bmatrix} 1 + w_{pc,d4} & 0 & -1 \end{bmatrix}^T & u_{pc,a} = \text{Heal AND } x_{pc,potion} \geq 1 \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} & u_{pc,a} = \text{Nothing OR Otherwise} \end{cases}$$
- The monsters state dependent action:

  $$f_{pc,a} = \begin{cases} \begin{bmatrix} -w_{mn,sf}(2 + w_{mn,d6}) & 0 & 0 \end{bmatrix}^T & \|x_{pc,p} - x_{mn,p}\|_1 \leq \text{Melee Range} \\ \begin{bmatrix} -w_{mn,sf}(2 + w_{mn,d8}) & 0 & 0 \end{bmatrix}^T & \text{Melee Range} < \|x_{pc,p} - x_{mn,p}\|_1 \leq \text{Ranged Range} \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T & \text{Ranged Range} < \|x_{pc,p} - x_{mn,p}\|_1 \end{cases}$$

The objective function and such is yet to be fully defined, but will consist of minimizing the monsters HP and maintaining the PC HP while also finding a way to minimize the time it takes to kill the monster.

1.1.4 Markov Implementation

For each of the stochastic functions defined in the update functions from the problem definition a markov chain can be described to better represent the changes between different states. Considering that only the actions are stochastic, the chains can be restricted to only the PC and Monster hp where the action input decides which of the chains are selected.

A few additional assumptions must be made to restrict the problem to a markov chain implementation:

- The monster and pc hp will be restricted to a finite range, $x_{pc,hp}, x_{mn,hp} \in \mathcal{X}_{hp} = \{0, 1, \ldots, n_{hp,max}\}$
- The ensuring of a potion will be a restriction upon the pc input/result in a nothing action if no potion is available.
- The deterministic movement will allow for monster action selection within closed-loop, however this splits the actual markov chain in two.
- The actual implimentation will keep the pc and monster hp independent of one another with actions being applied to the appropriate hp according to the action; however a proper markov chain would incorporate both into the states.

The states of the markov chain will be artificially independent $x_{pc,k}, x_{mn,k} = P(x_{pc,hp}),_(x_{mn,hp}) \in [0,1]^{n_{hp,max}+1}$. The transition matrices will be defined as $[p_{i,j}] = P(x_{k+1} = i, x_k = j)$. This can be thought of as each column being the probabilistic function results for $f(x_k, w_k), x_k = j, w_k = P[w_k]$.

Improve notation of this terrible thing...

PC Actions

First for the pc actions, which will be done independently from the monsters action which is then applied after according to the position.

When $u_k =$ Melee, the $f(x_k, w_k)$ will act upon $x_{mn,hp}$ with uncertainty $w_k \in \mathcal{W}_{d6}$, resulting in$f(x_{k+1} = i \; : \; x_k = j) = j - w_{pc,melee,sf} * (2 + w_{pc,d6}) = j - P(w_{pc,melee,sf} = [\begin{smallmatrix}0\\1\end{smallmatrix}]) \times (2 + w_{pc,d6} P(w_{pc,d6} = [\begin{smallmatrix}1 & \cdots & 6\end{smallmatrix}])) = j - [\begin{smallmatrix}0.45\\0.55\end{smallmatrix}] \times (2 + w_{pc,d6} [\begin{smallmatrix}1/6 & \cdots & 1/6\end{smallmatrix}]^T)$ resulting in $P(x_{k+1} = j) = 0.45$, $P(x_{k+1} = j - 2 - 1) = \cdots = P(x_{k+1} = j - 2 - 6) = \frac{0.55}{6} = \frac{11}{120} \approx 0.01967$. This is incorporated into the markov chain as the column j has

Notation terrible

$$\begin{array}{c} i = 1 \\ \vdots \\ i = j-2-7 \\ i = j-2-6 \\ \\ \vdots i = j-2-1 \\ i = j-2 \\ i = j-1 \\ i = j \\ i = j+1 \\ \vdots \end{array} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0.019677 \\ \vdots \\ 0.01967 \\ 0 \\ 0 \\ 0.45 \\ 0 \\ \vdots \end{bmatrix}$$

and the first few columns will have all the probabilities of negative values summed into the zero row. The markov chain transition matrix will be denoted as $P_{pc,melee}$.

confirm this direction... is this P or P^T?

Essentially the same definition is true with $u_k =$ ranged but with $w_{i,d8}$ and resulting in $P(x_{k+1} = j) = 0.45$, $P(x_{k+1} = j - 2 - 1) = \cdots = P(x_{k+1} = j - 2 - 8) = \frac{0.55}{8} = \frac{11}{160} = 0.06875$ and denoted with markov chain transition matrix $P_{pc,ranged}$.

Similarly for $u_k =$ heal but with $w_{i,d4}$ and not having the w_{sf}, meaning $P(x_{k+1} = j + 1 + 1) = \cdots = P(x_{k+1} = j + 1 + 4) = 1/4 = 0.25$ and denoted with markov chain transition matrix $P_{pc,heal}$.

Finally, for $u_k =$ nothing the transition matrix is defined as the identity matrix: $P_{pc,nothing} = \mathbf{I}_{n_{hp,max}}$

Monster Movement and Actions

For the simplistic case the movement is done first deterministically and then the monster actions are performed using the markov chains. Selection between the actions would be done deterministically based on position after movement and then the action is the same transition matrix for melee $P_{mn,melee}$, ranged $P_{mn,ranged}$, and nothing $P_{mn,nothing}$ are equivelent to the PC's transition matrices as $P_{pc,melee}$, $P_{pc,ranged}$, and $P_{pc,nothing}$ respectively.

The movement is done as described in the problem definition: $x_{k+1,mn,p} = x_{k,mn,p} + \text{direction}(x_{k+1,pc,p} - x_{k,mn,p})$.

The deterministic action input results in the monsters action being applied

$$P_{mn} = \begin{cases} P_{mn,melee} & \|x_{pc,p} - x_{mn,p}\|_1 \leq \text{Melee Range} \\ P_{mn,ranged} & \text{Melee Range} < \|x_{pc,p} - x_{mn,p}\|_1 \leq \text{Ranged Range} \\ P_{mn,nothing} & \text{Ranged Range} < \|x_{pc,p} - x_{mn,p}\|_1 \end{cases}$$

where each will act upon the PC hp.

Closed-loop Markov Chain

The actual markov chains associated with closed-loop will be selected directly based on state and player input by selecting the appropriate markov chains.

Since the assumption that u_k will be limited to those feasible by the PC, the following described as follows:

First, PC movement: $x_{k+1,pc,p} = x_{k,pc,p} + u_{pc,m}$.

being able to move after certaintly would add a lot more capability to the player's actions... however this would absolutely complicate it more...

Second, PC action:

$$\begin{bmatrix} x_{k+1,pc,hp} \\ x_{k+1,mn,hp} \end{bmatrix} = \begin{cases} \begin{bmatrix} x_{k,pc,hp} \\ x_{k,mn,hp}^T P_{pc,melee} \end{bmatrix} & u_k = \text{melee} \\ \begin{bmatrix} x_{k,pc,hp} \\ x_{k,mn,hp}^T P_{pc,ranged} \end{bmatrix} & u_k = \text{ranged} \\ \begin{bmatrix} x_{k,pc,hp}^T P_{pc,heal} \\ x_{k,mn,hp} \end{bmatrix} & u_k = \text{heal} \\ \begin{bmatrix} x_{k,pc,hp} \\ x_{k,mn,hp} \end{bmatrix} & u_k = \text{nothing} \end{cases}$$

Third, monster movement: $x_{k+1,mn,p} = x_{k,mn,p} + \text{direction}(x_{k+1,pc,p} - x_{k,mn,p})$.

Finally, monster action:

$$x_{k+1,pc,hp} = \begin{cases} x_{k,pc,hp}^T P_{mn,melee} & \|x_{k+1,pc,p} - x_{k+1,mn,p}\|_1 \leq \text{Melee Range} \\ x_{k,pc,hp}^T P_{mn,ranged} & \text{Melee Range} < \|x_{k+1,pc,p} - x_{k+1,mn,p}\|_1 \leq \text{Ranged Range} \\ x_{k,pc,hp} & \text{Ranged Range} < \|x_{k+1,pc,p} - x_{k+1,mn,p}\|_1 \end{cases}$$

It is possible to incorporate this into a single set of markov chains in higher dimensions; however, this sequential definition seems to make the most sense instead of defining a finite number of position states and closing the loop entirely.

1.2 MATLAB implimentation

1.2.1 Additional Simplifications

Due to the nature of the problem being of such high dimension, the following additional simplifications will be done:

- The policy will be only based on relative distances between the PC and Monster. (i.e.)

$$x_{k,pos} = x_{pc,p} - x_{mn,p} = \begin{bmatrix} x_{pc,p,x} - x_{mn,p,x} \\ x_{pc,p,y} - x_{mn,p,y} \end{bmatrix}$$

  This is mainly to reduce the number of dimensions from 4 to 2 and ensure the finite position is useful.

- The objective function will be only based on the relative HP between PC and monster, $x_{pc,hp} - x_{mn,hp}$. Specifically, the stage-cost function will be to maximize this relative HP:

$$g_k(x_{pc}, x_{mn,hp}) = -(x_{pc,hp} - x_{mn,hp})$$

- A value iteration method will be used (first attempted in Infinite-Horizon... but might be a finite horizon instead).
- When determining the policy, the cost-function will be augmented to discourage moving too far away and a strong discouragement of losing (or encouragement of winning) will be included.

1.2.2 Optimal Policy

Update Probabilities

The probabilites fore each state update (essentially a large-multi dimensional markov-chain) will be computed ahead of time to make the policy optimization more computationally efficient. Specifically the closed-loop markov chains will be used to determine the probabilities of future states $\mathbf{P}[x_{k+1}]$ that can then be used for the expected next time-step cost computation:

$$\mathbf{E}[J(f(x_k, u_k, w_k))] = \mathbf{P}[x_{k+1}]^T J_{k+1}(x_{k+1})$$

Value Iteration

The optimal policy will be determined for every relative position and the current HPs. These states will be treated as within a 4-D grid which are looped through (potentially in parallel for computation purposes... or mabye a vector version can be determined...) and the optimal policy will be updated each iteration.

This could probably instead be done with a much better set of matrix/vector operations but that isn't yet figured out...

this could be explained much better... but essentially use the code from previous examples...