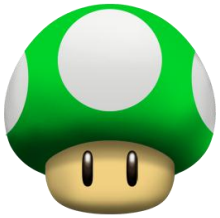
INF 1771 – Inteligência Artificial

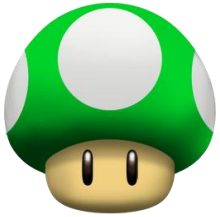
Aula 26 – Máquinas de Estados Finitos

Edirlei Soares de Lima
<elima@inf.puc-rio.br>



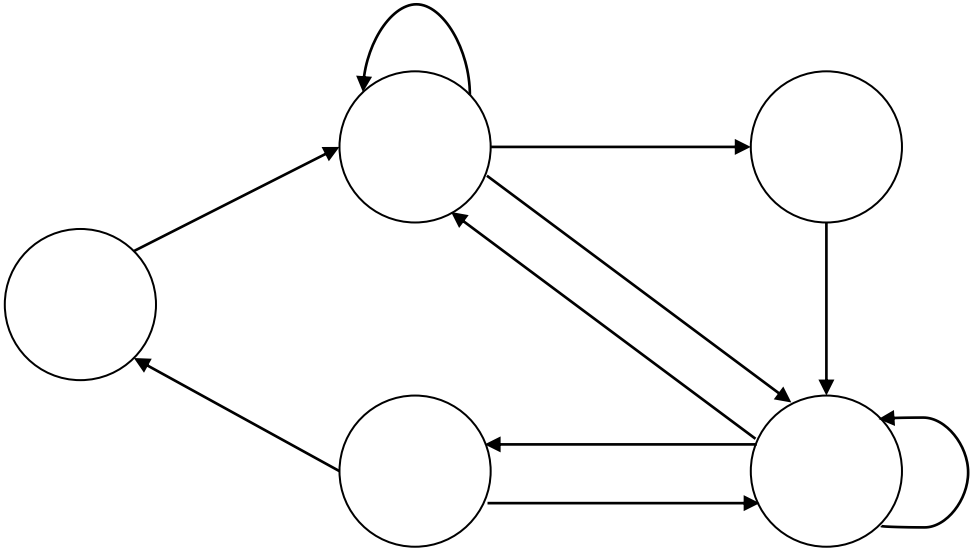
Introdução

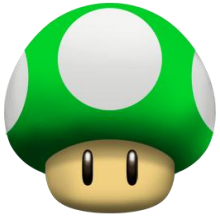
- **Máquinas de Estados Finitos** (Finite State Machines - FSM) são provavelmente o padrão de software mais utilizado em jogos para selecionar o comportamento de agentes reativos.



Máquina de Estados

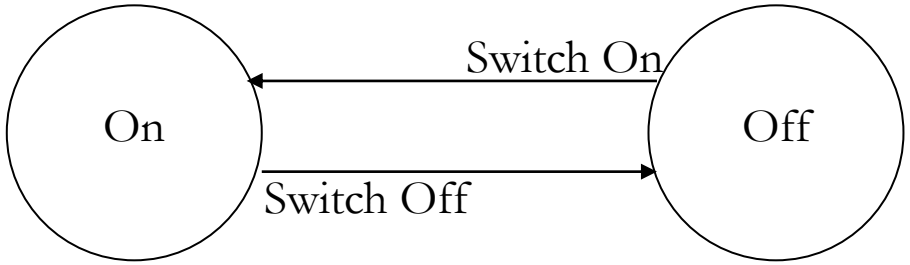
- Uma **máquina de estados** é um modelo matemático usado para representar programas.
  - Conjunto de estados.
  - Regras de transição entre estados.
  - Estado atual.



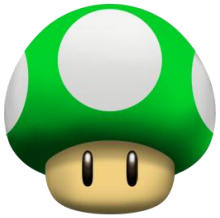


Máquina de Estados

- Um exemplo bem simples de uma FSM é um interruptor de luz.

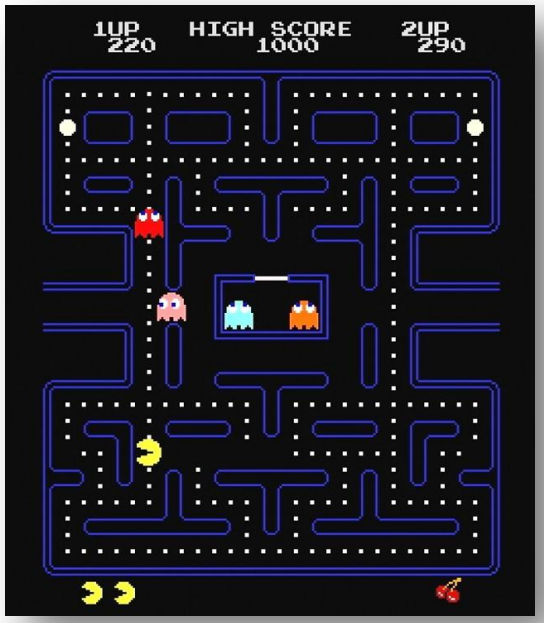


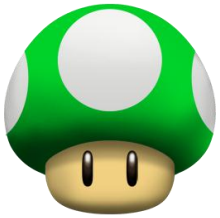
- Em um jogo normalmente uma FSM não é tão simples assim, visto que geralmente os agentes podem ter um **conjunto muito maior de estados**.



Exemplo – Pac-Man

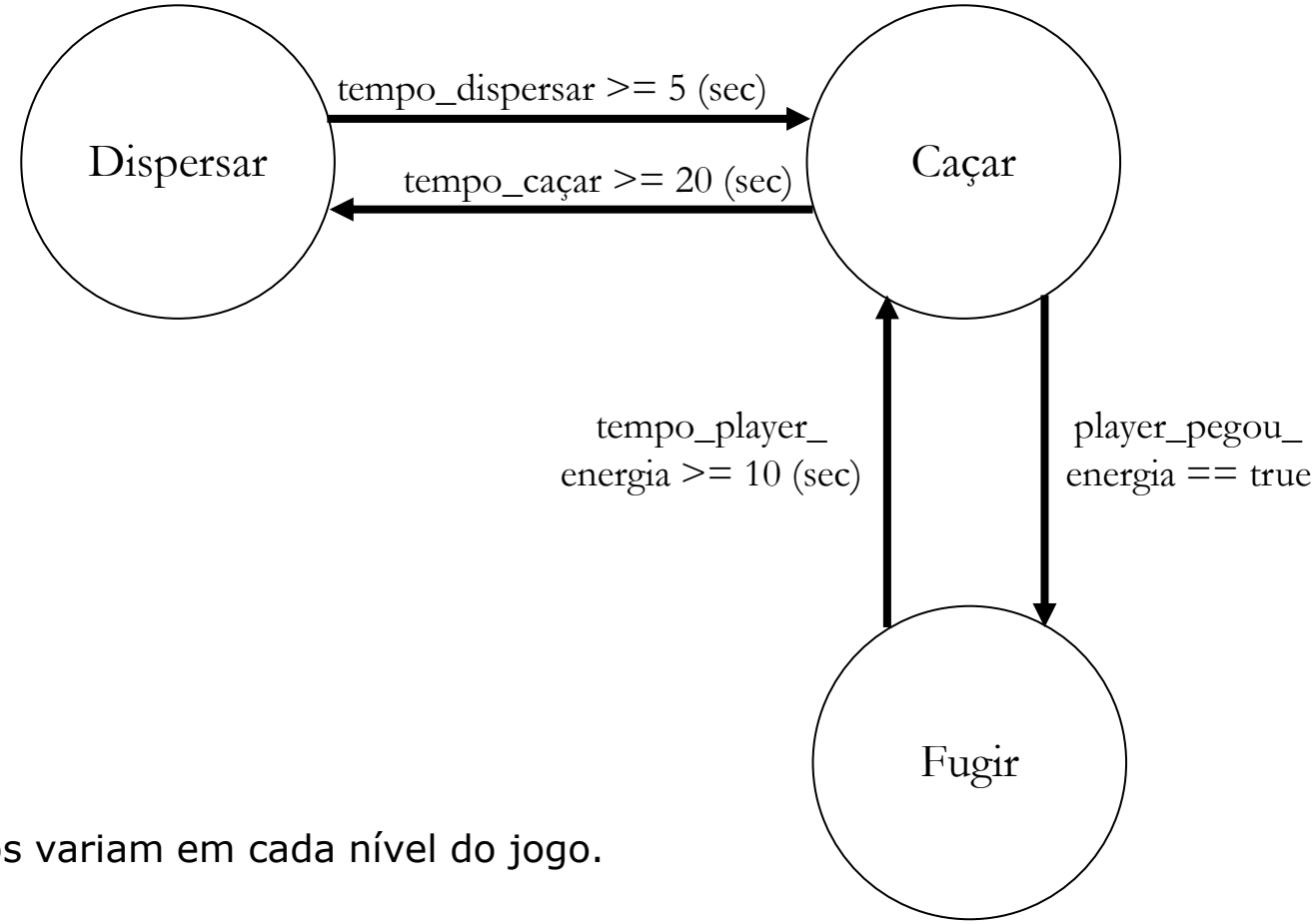
- Os fantasmas Inky, Pinky, Blinky e Clyde do jogo **Pac-man** são implementados via FSM.
- Os fantasmas tem **3 comportamentos**:
  - Caçar (Chase)
  - Fugir (Evade)
  - Dispersar (Scatter)
- A **transição** de estados ocorre sempre que o jogador conseguir alguma pílula de energia.
- A implementação da **ação caçar** de cada fantasma é diferente.



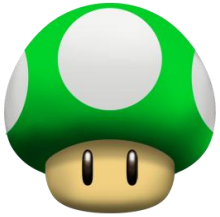


Exemplo – Pac-Man

- Máquina de Estados:

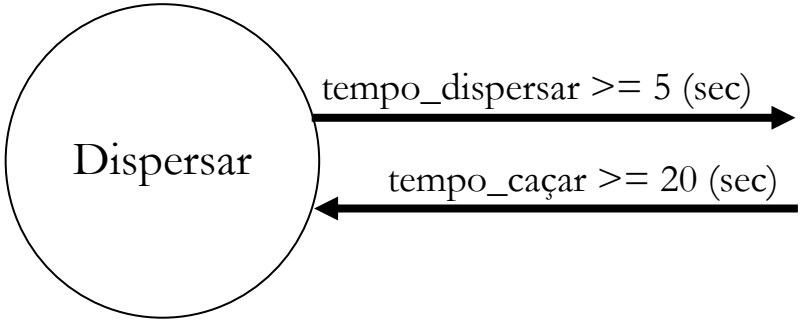


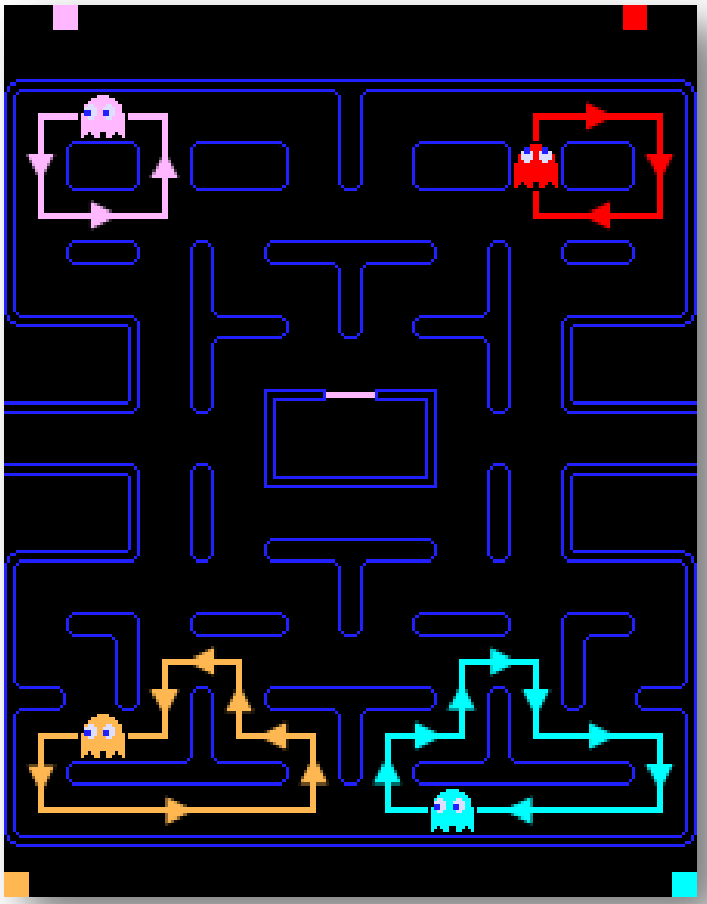
- Os tempos variam em cada nível do jogo.

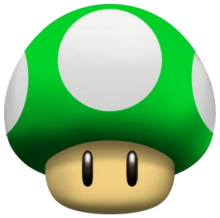


Exemplo – Pac-Man

- **Comportamento de Dispersar**:
  - Mover em direção aos cantos e ficar andando em círculos.

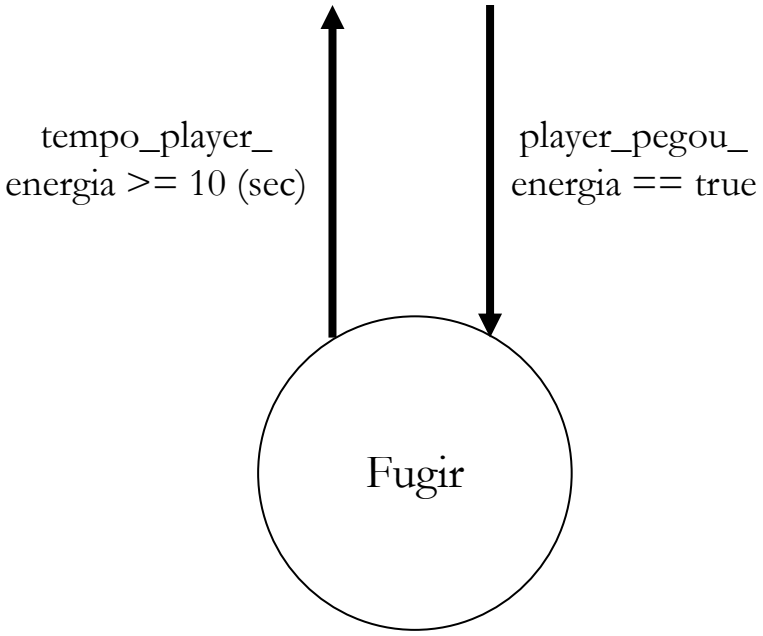


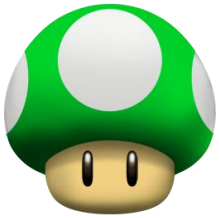




Exemplo – Pac-Man

- **Comportamento de Fugir**:
  - Movimentar-se mais lentamente com movimentos aleatórios.

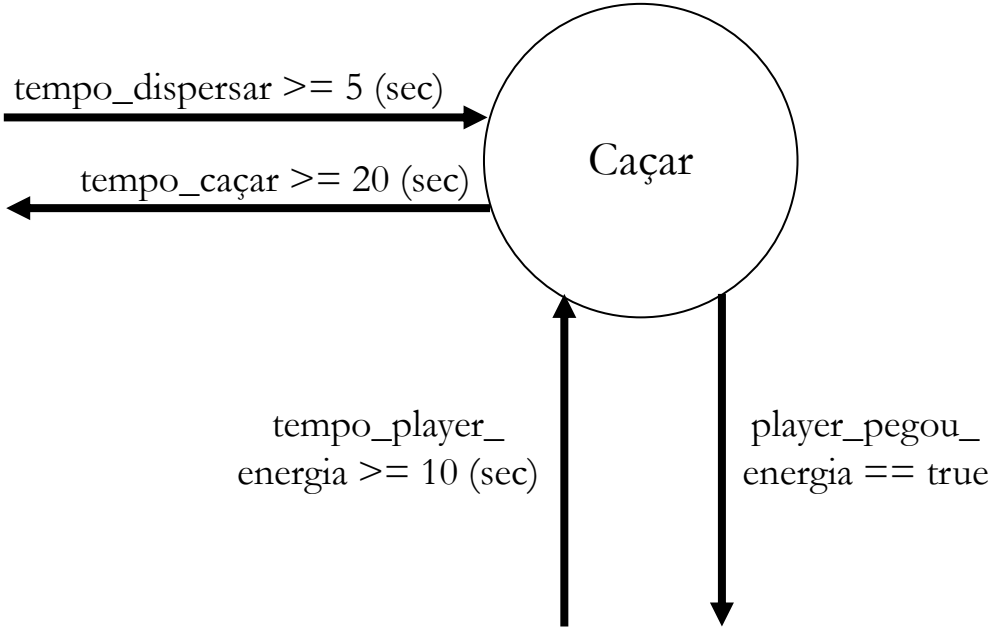


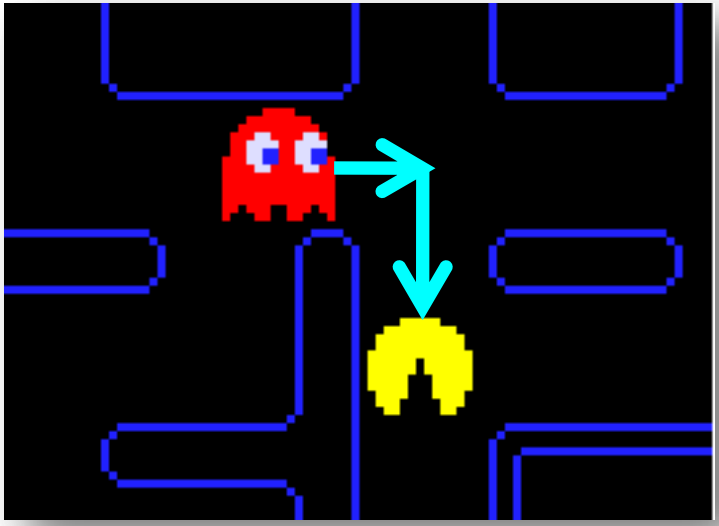


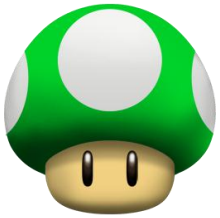
Exemplo – Pac-Man

- **Comportamento de Caçar**:

  - Movimenta-se mirando na posição do Pac-Man.



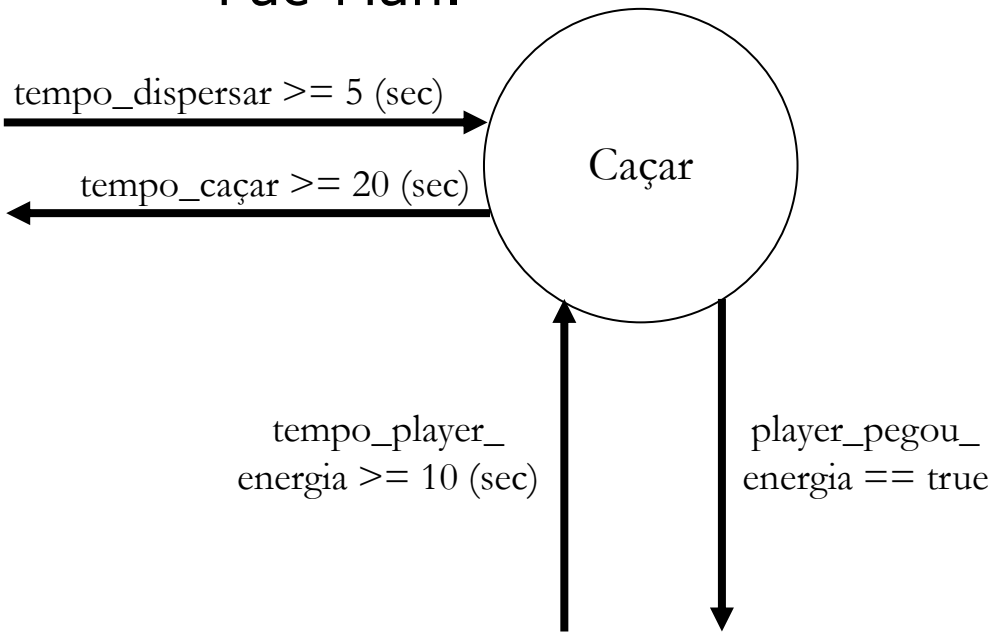


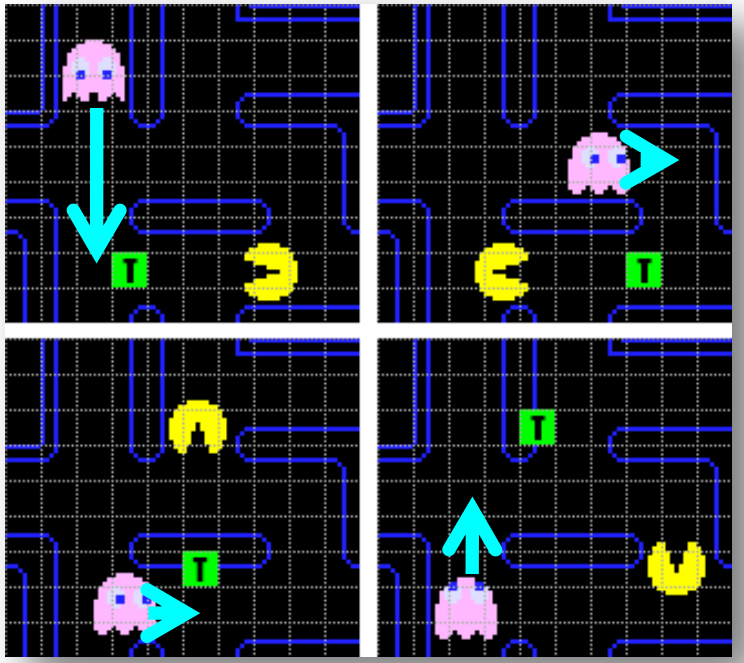


Exemplo – Pac-Man

- **Comportamento de Caçar**:

  - Movimenta-se mirando na posição 4 tiles a frente do Pac-Man.

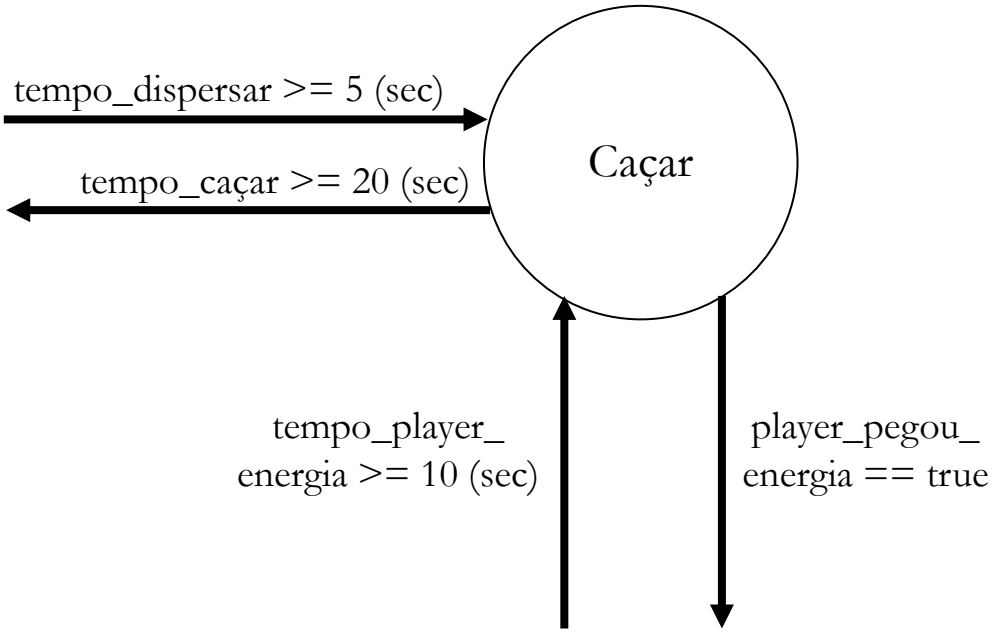


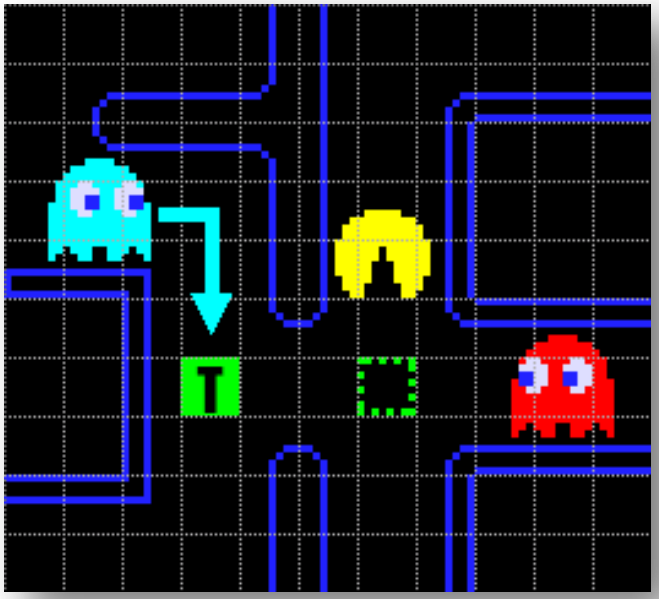


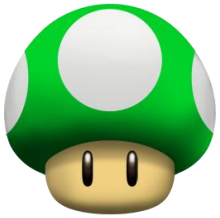
Exemplo – Pac-Man

- **Comportamento de Caçar**:

  - Movimenta-se mirando em uma posição que combina a posição/direção do Pac-Man e do Blinky.



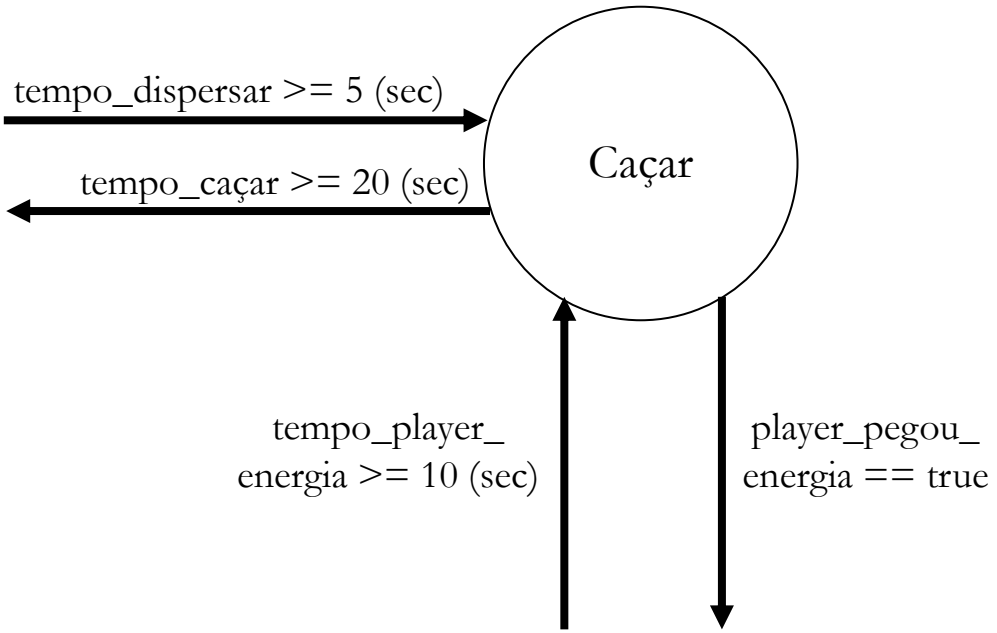


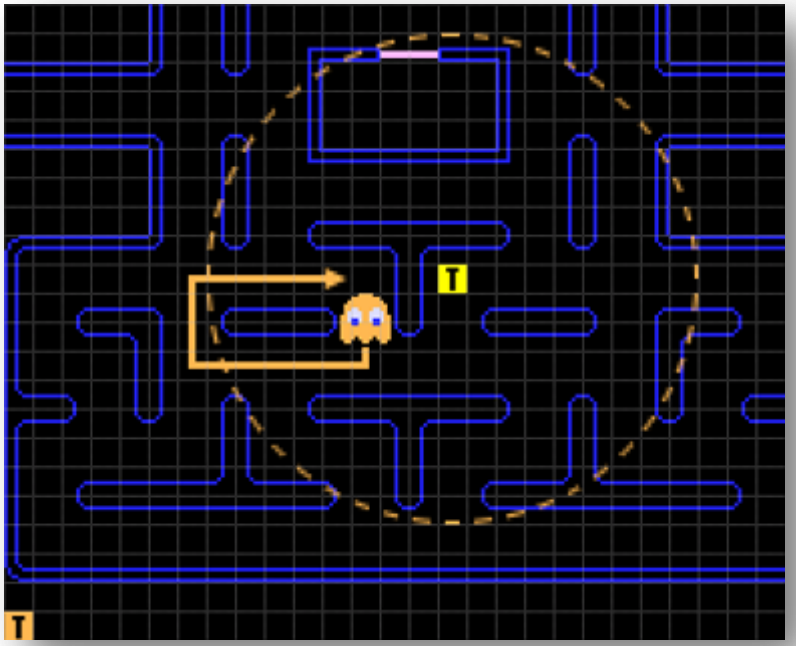


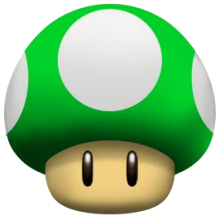
Exemplo – Pac-Man

- **Comportamento de Caçar**:

  - Quando está longe do Pac-Man, movimenta-se em direção ao Pac-Man. Quando está perto, movimenta-se em direção ao canto da tela.

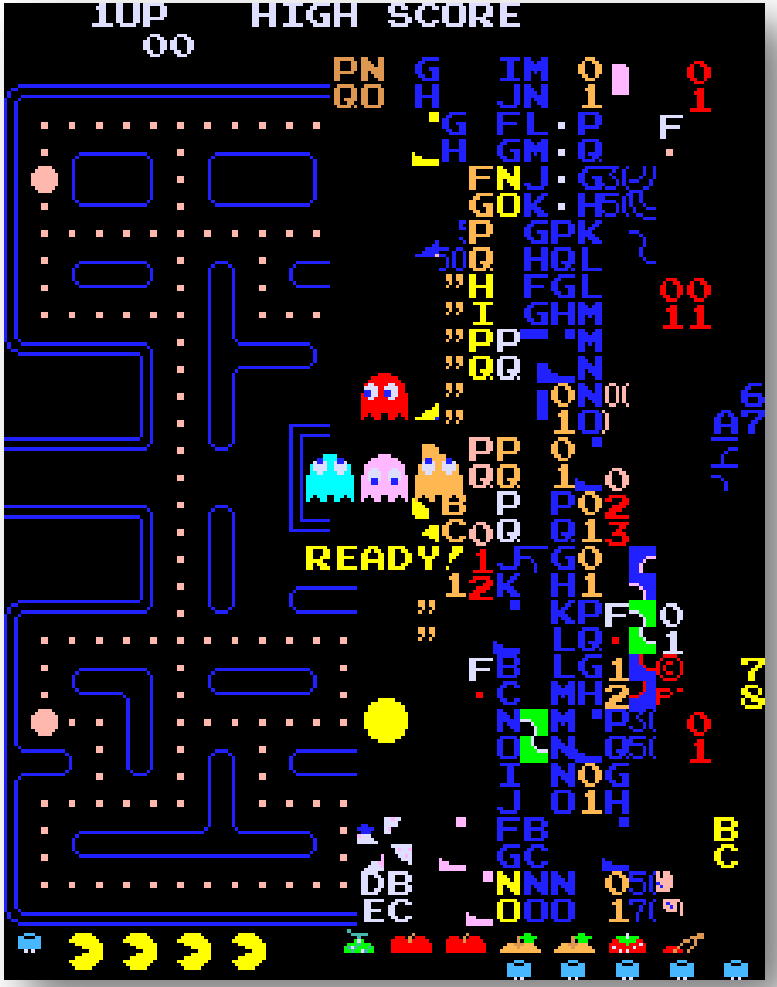


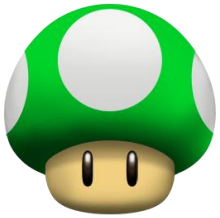




Exemplo – Pac-Man

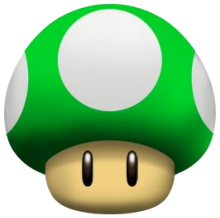
- Fim de jogo?
- Teoricamente Pac-Man foi projetado para não ter fim, mas… no level 256…





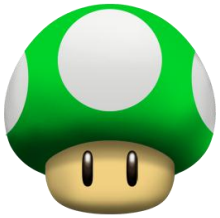
Exemplo – Quake

- Os NPCs do jogo **Quake** também são implementados via FSM.
- **Estados/Comportamentos**:
  - Procurar Armadura (FindArmor)
  - Procurar Kit Medico (FindHelth)
  - Correr (RunAway)
  - Atacar (Attack)
  - Perseguir (Chase)
  - ...
- Até mesmo as armas são implementadas como uma mini FSM.
  - Mover (Move)
  - Tocar Objeto (TouchOject)
  - Morrer/Explodir (Die)

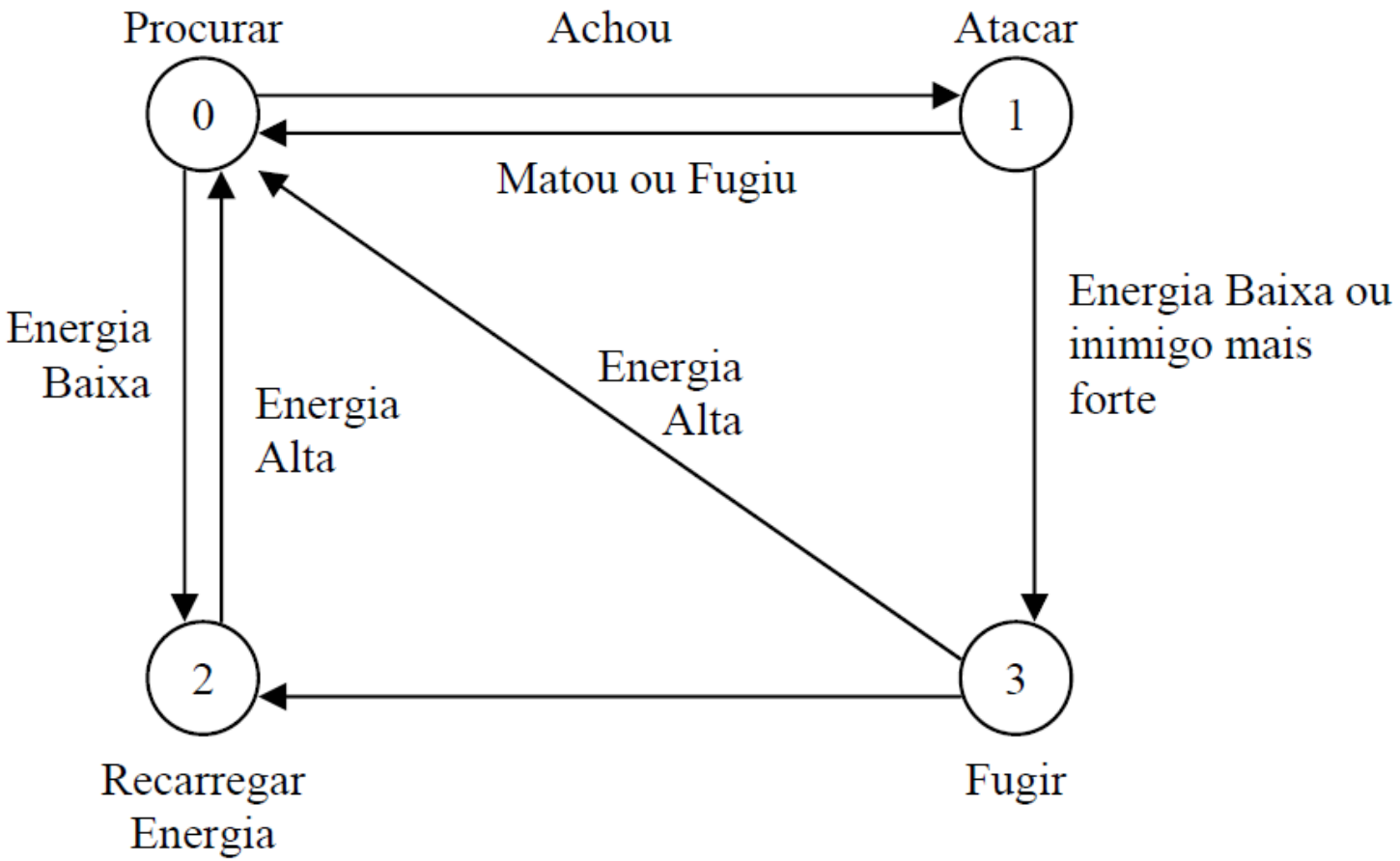


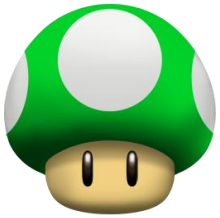
Exemplo – FIFA20XX

- O comportamento dos jogadores é definido através de FSMs.
- **Estados/Comportamentos**:
  - Driblar (Dribble)
  - Correr Atrás da Bola (ChaseBall)
  - Marcar Jogador (MarkPlayer)
  - ...
- Os times também usam FSMs para definir comportamentos em grupo.
  - Defender (Defend)
  - Atacar (Attack)
  - ...



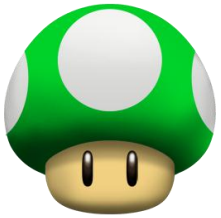
Máquina de Estados





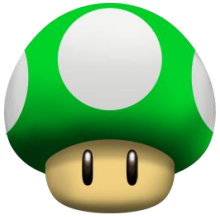
Implementação

```
void run(int *state){
  switch(*state){
    case 0: //procurar inimigo
      procurar();
      if(encontrou_inimigo)
        *state = 1;
      break;
    case 1: //atacar inimigo
      atacar();
      if (morto){
         morrer();
         *state = -1;
      }
      if (matou){
         *state = 0;
      }
      if(energia < 50 || inimigo_forte)
         *state = 3;
     break;
    case 2: //recarregar energia
      recarregar();
      if(energia > 90)
        *state = 0;
    break;
    case 3: //fugir
      fugir();
      if(!encontrou_inimigo){
        if(energia < 50)
          *state = 2;
        else
          *state = 0;
       }
       break;
   }
}
```



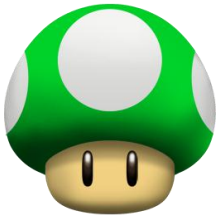
Vantagens

- **Elas são rápidas e simples de implementar –** existem várias formas de implementar e todas são muito simples.
- **Gastam pouco processamento.**
- **São fáceis de depurar** – quando o numero de estados é pequeno.
- **São intuitivas** – qualquer pessoa consegue entender o seu significado apenas olhando para a sua representação visual. Isso facilita o trabalho do **game designer**, que muitas vezes não tem conhecimento de linguagens de programação.
- **São flexíveis** – podem ser facilmente ajustada pelo programador para prover comportamentos requeridos pelo game designer.

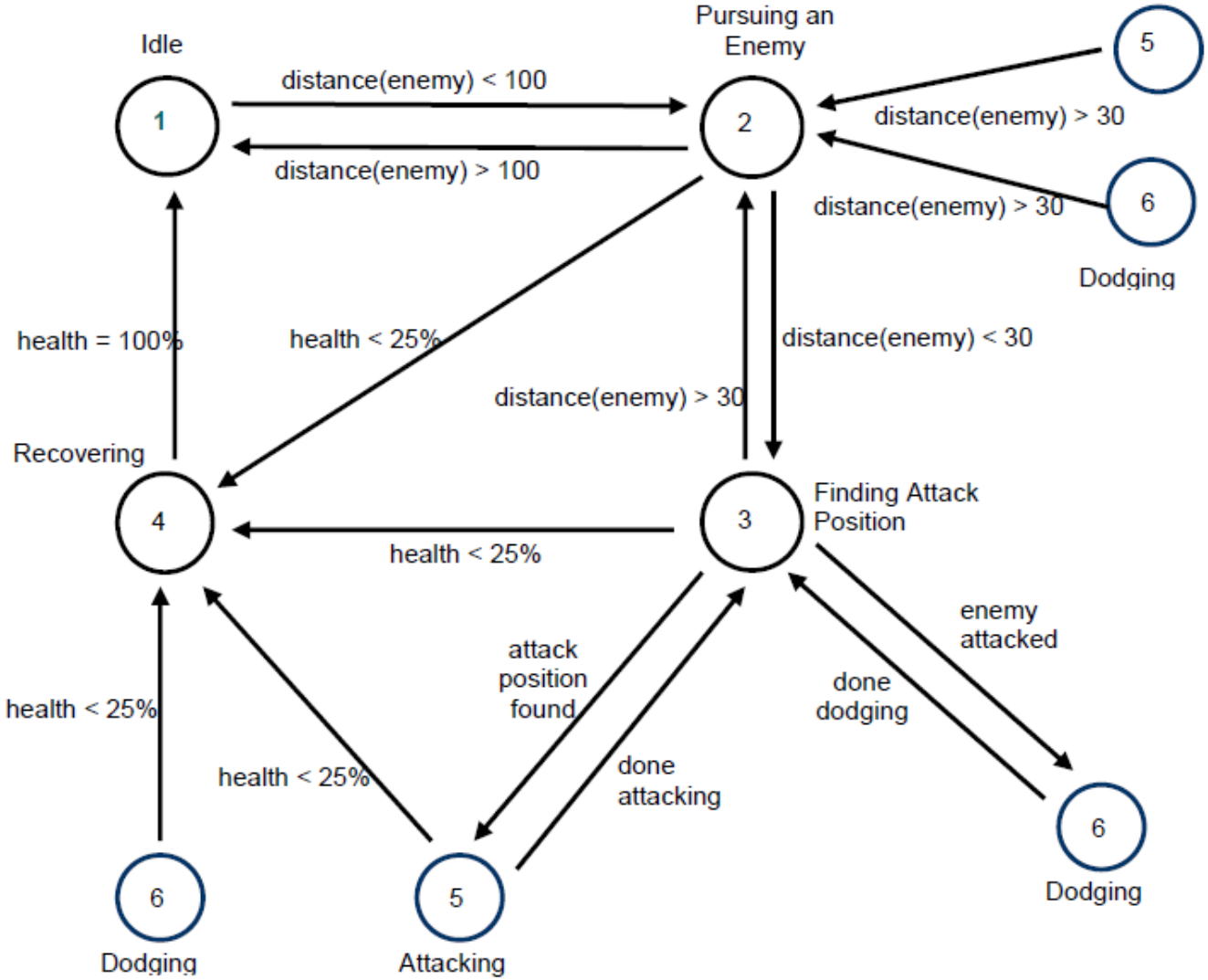


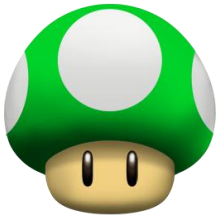
Problemas

- Á medida que a complexidade do comportamento dos agentes aumenta, as FSMs tendem a **crescer de forma descontrolada.**
- As FSMs se tornam terrivelmente complexas quanto levam em consideração **ações muito básicas** necessárias a um agente.
- A **representação visual** torna-se intratável.
- Comportamentos complexos **são necessários em jogos modernos**.



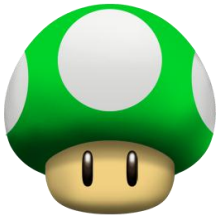
FSM um Pouco Mais Complexa...



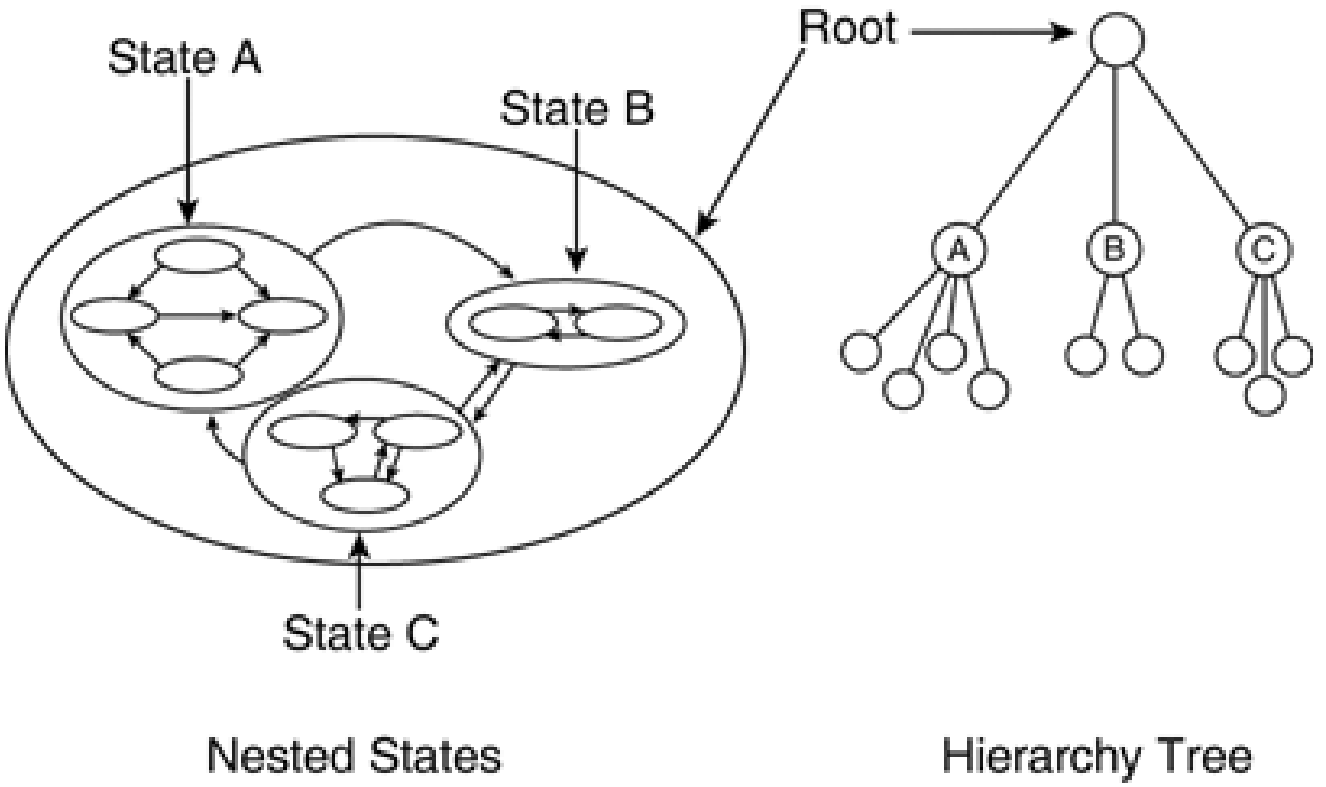


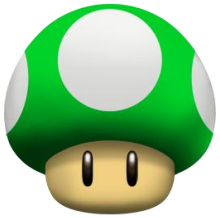
Máquina de Estados Finita Hierárquica

- É possível organizar uma FSM usando **máquinas de estados finitas hierárquicas** (HFSM).
- Níveis mais altos lidam com ações mais genéricas, enquanto níveis mais baixos lidam com ações mais específicas.
- **Cada estado pode ser uma nova FSM**.
- Infelizmente, a hierarquia não reduz o número de estados. Ela pode somente pode reduzir significativamente o número de transições e tornar a FSM **mais intuitiva e simples de compreender**.



Máquina de Estados Finita Hierárquica





Máquina de Estados Finita Hierárquica