Modelagem de Saturação de Defesa Aérea e Negação de Área (A2/AD): Uma Abordagem Vetorial Baseada em Autômatos Celulares

Rafael Peralta Góes

RESUMO

A saturação de sistemas de defesa aérea por enxames de veículos não tripulados e aeronaves de alta performance constitui um dos principais desafios da guerra moderna, ameaçando a integridade de zonas de Negação de Área (A2/AD). Neste trabalho, é apresentado um modelo computacional baseado em autômatos celulares de fluxo, capaz de simular a interação dinâmica entre uma força atacante heterogênea (Drones, Caças e Stealth) e um sistema de defesa degradável. O modelo utiliza uma matriz bidimensional para representar o espaço aéreo, onde a dinâmica de transição incorpora regras de movimentação vetorial, probabilidades de interceptação e a mecânica de Supressão de Defesas Aéreas Inimigas (SEAD), permitindo que unidades em recarga sejam destruídas. Os resultados obtidos demonstram a existência de um limiar crítico de densidade defensiva, abaixo do qual o sistema sofre um colapso estrutural não-linear devido à fadiga de recarga. Este modelo oferece uma ferramenta flexível para a compreensão de táticas de saturação e dimensionamento de redes defensivas.

1. INTRODUÇÃO

O conceito de Anti-Acesso e Negação de Área (A2/AD) depende da capacidade de um sistema defensivo impor um custo proibitivo à operação inimiga. No entanto, o advento de táticas de enxame (*swarming*) e o uso massivo de drones de baixo custo buscam exaurir os interceptadores disponíveis, explorando os tempos de recarga dos sistemas de mísseis superfície-ar (SAM).

Ferramentas de modelagem computacional, especificamente os Autômatos Celulares (AC), têm se mostrado essenciais para o estudo de sistemas complexos onde comportamentos emergentes surgem de regras locais simples[2]. Diferente dos modelos de equações diferenciais clássicos, os ACs permitem simular a heterogeneidade espacial e eventos estocásticos, como a falha de um radar ou a penetração de uma aeronave furtiva.

O presente trabalho apresenta um modelo de autômato celular vetorial para simular a saturação de defesa, com ênfase na mecânica de recarga e supressão (SEAD). O modelo avalia diferentes densidades de defesa para identificar o "ponto de ruptura" onde a integridade do espaço aéreo é perdida.

2. METODOLOGIA

O modelo é implementado em Python, utilizando a biblioteca NumPy para processamento vetorial de alta performance e Pygame para visualização. A grade de simulação é uma matriz bidimensional onde cada célula pode assumir estados discretos que representam o vazio, elementos de defesa ou ameaças aéreas:

- **Estado 0 (Vazio):** Espaço aéreo livre.
- **Estado 2 (SAM Pronta):** Bateria antiaérea ativa e capaz de engajar alvos.
- **Estado 3 (SAM Recarga):** Bateria em processo de remuniciamento, vulnerável à destruição.
- **Estado 4 (Interferência):** Destroços ou contramedidas (*Chaff*) que bloqueiam a linha de visada.
- **Estados 5, 6, 7 (Ameaças):** Representam Drone, Caça e Stealth, respectivamente.

A simulação opera em fluxo contínuo (norte para sul) e incorpora taxas de regeneração (recarga) e decaimento (dissipação de interferência).

2.1 Regras de Transição

As regras são aplicadas sincronicamente a cada geração (tick), utilizando vizinhança estendida (Moore):

1. **Fluxo de Ataque:** Inimigos movem-se do topo para a base da matriz. Diferentes classes possuem capacidades distintas: *Caças* podem avançar duas células por turno (velocidade supersônica), enquanto *Stealths* possuem baixa assinatura de radar.
2. **Interceptação e Combate:** Uma célula "SAM Pronta" engaja inimigos em sua vizinhança. A probabilidade de abate varia conforme a classe do alvo (ex: 20% de chance de detectar Stealth). O abate gera uma célula de "Interferência" que cega temporariamente as defesas adjacentes.
3. **Ciclo de Recarga:** Após disparar, uma SAM transita para o estado "Recarga" por 15 gerações.
4. **Supressão de Defesa (SEAD):** Uma inovação deste modelo é a regra de atropelamento: se uma aeronave inimiga tenta mover-se para uma célula contendo uma "SAM em Recarga", a defesa é destruída e o inimigo ocupa a posição. Isso simula a destruição física de lançadores vulneráveis.

2.2 Implementação Computacional e Algoritmos

A simulação foge da abordagem tradicional de iterar sobre cada célula com laços de repetição (*loops*), o que seria computacionalmente custoso para grades grandes. Em vez disso, optou-se por uma abordagem **vetorizada** utilizando a biblioteca *NumPy*. O estado global do sistema é manipulado através de operações matriciais simultâneas, garantindo alta performance e sincronicidade nas transições de estado.

2.2.1 Cálculo de Vizinhança via Deslocamento Matricial :

Para determinar a vizinhança de Moore (8 vizinhos) de cada célula sem percorrer a matriz, utilizou-se a função `numpy.roll`. Esta operação desloca todos os elementos da grade em uma direção, permitindo alinhar os vizinhos "futuros" com a posição "atual" para comparação lógica.

O código abaixo ilustra como a ameaça das baterias SAM é propagada instantaneamente para as células adjacentes:

```
sam_n = np.roll(sams_letais, 1, axis=0)
sam_s = np.roll(sams_letais, -1, axis=0)
sam_w = np.roll(sams_letais, 1, axis=1)
sam_e = np.roll(sams_letais, -1, axis=1)

ameaca_sam = sam_n | sam_s | sam_w | sam_e
```

Figura 1: *Cálculo de Vizinhança via Deslocamento Matricial.*

2.2.2 Lógica de Movimento e Supressão (SEAD) :

A movimentação dos agentes (Drones, Caças) é resolvida através de **Máscaras Booleanas**. Uma máscara lógica identifica quais agentes desejam mover-se e uma segunda máscara verifica se o destino é válido.

A mecânica de Supressão de Defesas (SEAD), onde o inimigo destrói uma SAM em recarga, é implementada permitindo que o estado `SAM_RECARGA` seja considerado um destino válido ("livre") para o movimento, resultando na sobrescrita do valor da célula:

```
# O destino é válido se for VAZIO, FUMAÇA ou SAM_RECARGA (Amarela)
# Isso permite que o inimigo "atropele" a SAM em recarga.
mask_destino_livre = (destino_potencial == VAZIO) | \
                     (destino_potencial == INTERFERENCIA) | \
                     (destino_potencial == SAM_RECARGA)

quem_move = mask_quem & mask_destino_livre
nova_grade[quem_move] = VAZIO
mask_destino = np.roll(quem_move, steps, axis=0)
return quem_move, mask_destino
```

Figura 2: *Lógica de Movimento e Supressão (SEAD).*

2.1.3 Coleta de Dados A simulação opera em um laço principal (*game loop*) controlado pela biblioteca *Pygame*, que renderiza a matriz a 15 quadros por segundo (FPS). A coleta de dados estatísticos ocorre ao final de cada ciclo de processamento, antes da renderização, contabilizando:

1. Soma de células de ameaça ativas (`np.sum`).
2. Contagem de eventos de abate retornados pela função de transição.
3. Detecção de vazamento na última linha da matriz (`grade[-1, :]`).

---

3. RESULTADOS

Os resultados gerados pelas simulações revelaram comportamentos não-lineares críticos, dependendo da densidade inicial de baterias SAM. A análise baseia-se nas curvas de "Inimigos Ativos" (carga do sistema) e "Vazamento Acumulado" (falhas de defesa).

1. Colapso Imediato (Densidade 10%)

Com baixa densidade de defesa (10%), o sistema entra em colapso quase instantaneamente. Como observado na Figura 1, a taxa de vazamento (linha verde) cresce linearmente a partir da geração 50. A curva de inimigos ativos (azul) apresenta crescimento contínuo, indicando que os atacantes estão se acumulando no espaço aéreo, pois as defesas são destruídas (via regra SEAD) antes que possam completar o ciclo de recarga.

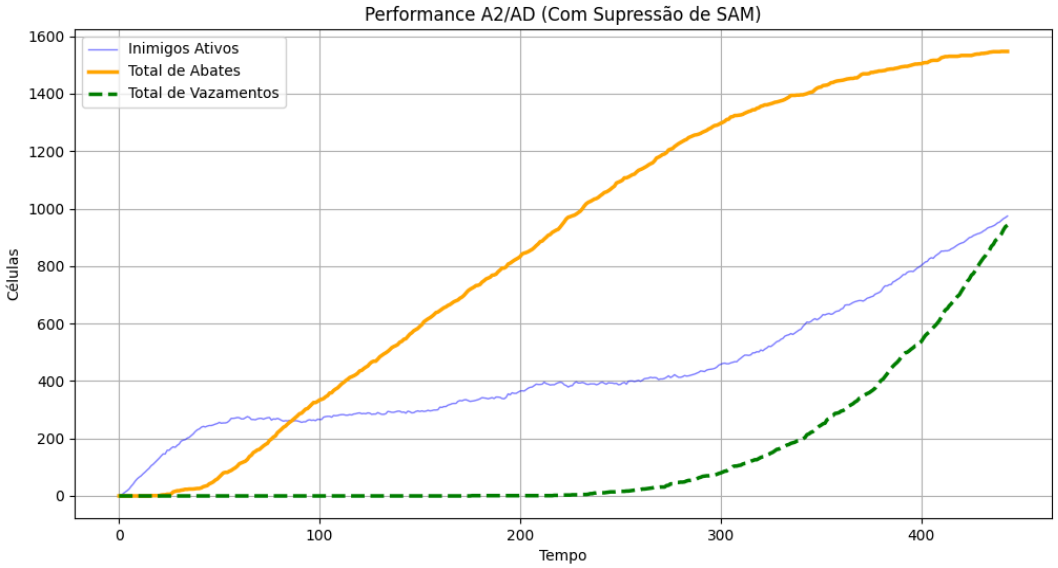

Figura 1: *Cenário de baixa densidade (0.10). Nota-se o crescimento descontrolado de inimigos ativos e vazamento constante.*

---

2. Ponto de Ruptura (Densidade 15%)

Este cenário ilustra o fenômeno de "fadiga de sistema". Conforme a Figura 2, a defesa mantém a integridade (vazamento zero) até a geração 300. Contudo, a acumulação probabilística de perdas de SAMs em recarga cria uma brecha catastrófica entre as gerações 350 e 400. O sistema transita subitamente de um estado de controle para um estado de saturação total, triplicando o número de inimigos ativos.

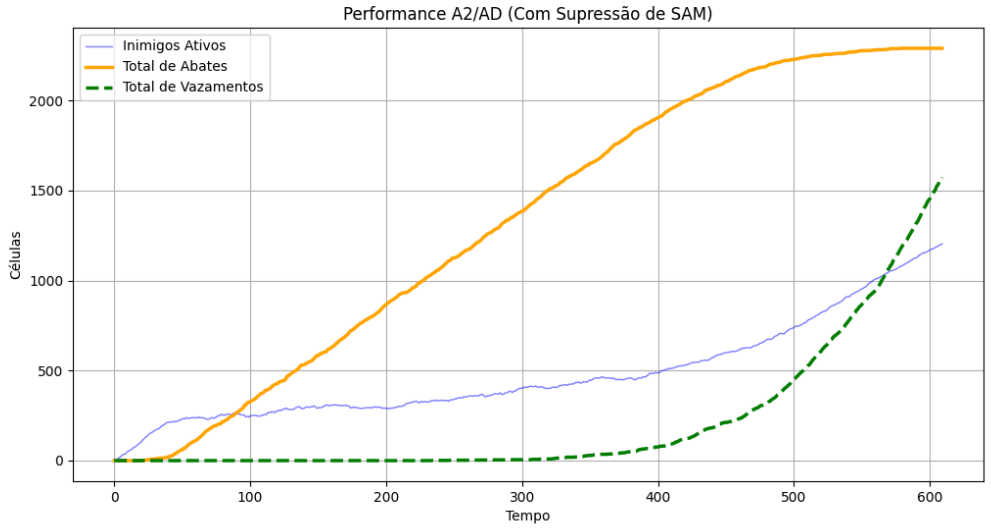

Figura 2: *Cenário de transição (0.15). O colapso ocorre tardiamente (Geração ~350), demonstrando a não-linearidade da falha defensiva.*

---

3. Resistência Elástica (Densidade 25%)

Com 25% de cobertura (Figura 3), o sistema demonstra robustez. A redundância de baterias permite que SAMs vizinhas cubram unidades em recarga. O vazamento é iniciado apenas tardiamente (Geração 650) e cresce de forma lenta e gradual, indicando que o sistema ainda impõe atrito significativo ao atacante, perdendo a integridade apenas por exaustão a longo prazo.

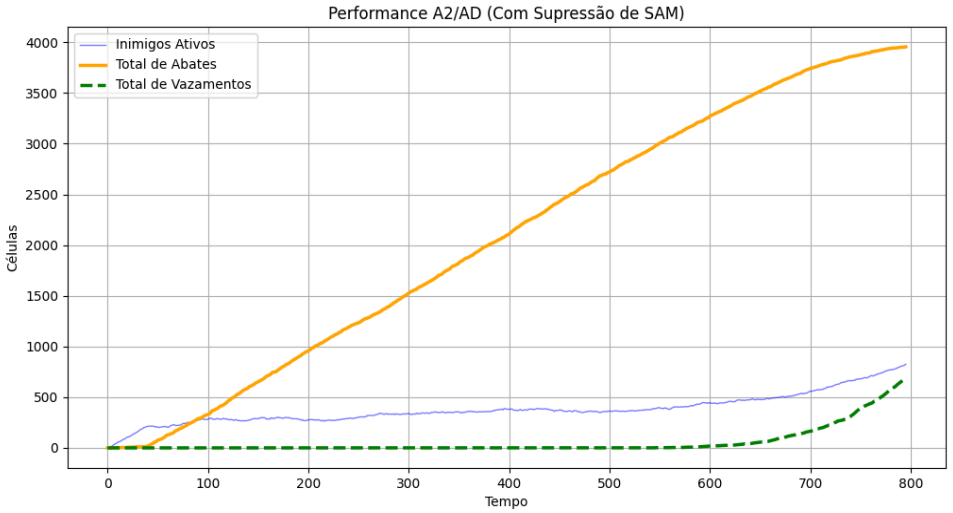

Figura 3: Cenário de resistência (0.25). O vazamento é residual e tardio.

---

4. Negação de Área Total (Densidade 50%)

No cenário de alta densidade (Figura 4), observa-se a supremacia defensiva. A linha de vazamento permanece próxima de zero. A redundância é tão alta que a vulnerabilidade do tempo de recarga é anulada pela presença de outras baterias prontas. O inimigo não consegue estabelecer corredores de penetração.

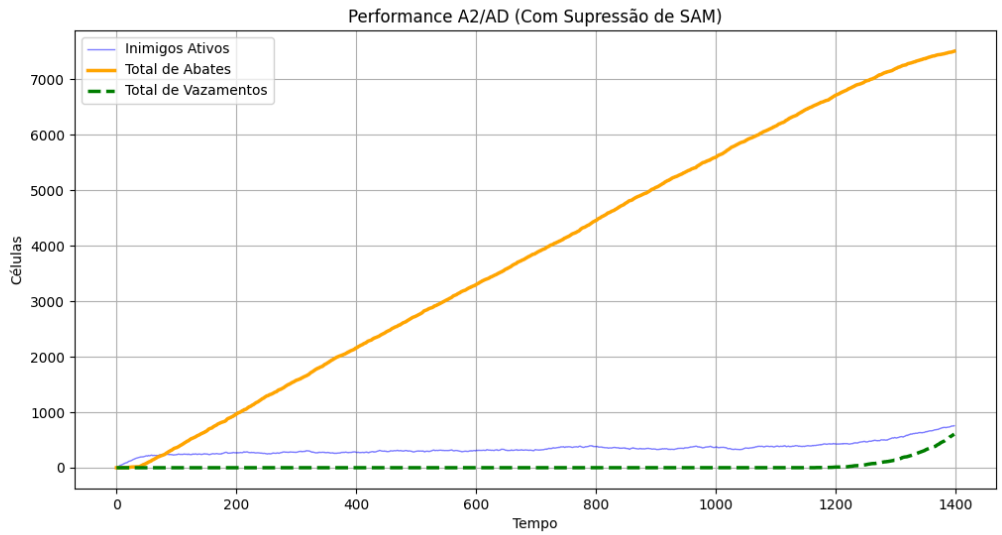

Figura 4: *Cenário de Negação de Área (0.50). Eficácia total com vazamento nulo.*

---

4. DISCUSSÃO

O modelo confirma que a eficácia de uma rede A2/AD não escala linearmente com o número de lançadores, mas apresenta comportamentos de limiar (*threshold*). A introdução da vulnerabilidade durante a recarga provou ser o fator determinante para o colapso: uma vez que a taxa de destruição de defesas supera a taxa de recuperação (recarga), o sistema entra em uma espiral de falha irreversível, análoga ao processo de degradação ambiental observado em modelos ecológicos[3].

Em contextos reais, isso sugere que a sobrevivência de um sistema antiaéreo depende mais da proteção mútua (sobreposição de campos de tiro) do que da capacidade individual de cada bateria.

5. CONCLUSÃO

O modelo apresentado oferece uma abordagem visual e quantitativa para explorar conceitos de guerra assimétrica e saturação. Ao utilizar um autômato celular vetorial com estados complexos (recarga, interferência, tipos de aeronaves), a simulação demonstrou como interações locais geram padrões globais de ruptura defensiva. A identificação do "Ponto de Ruptura" (observado em 15% de densidade) é crítica para o planejamento de defesa, indicando o mínimo necessário para evitar a falha catastrófica sob ataque contínuo.

REFERÊNCIAS

1. Bagagi, A. S. *Modelagem de Propagação de Agricultores e Degradação do Solo*.
2. Ilachinski, A. *Cellular Automata: A Discrete Universe*. World Scientific (2001). [5]
3. NumPy Developers. NumPy Documentation. Disponível em: https://numpy.org/. [6]
4. Matplotlib Developers. Matplotlib Documentation. Disponível em: https://matplotlib.org/. [7]