RELATÓRIO TÉCNICO

Sistema de Controle de Tráfego Aéreo

Projeto Margolis

Autores: Guilherme Ganassini, Gustavo Domenech
Disciplina: Sistemas Operacionais
Data: Agosto de 2025

Contents

1. Resumo Executivo

O projeto Margolis implementa um simulador de controle de tráfego aéreo utilizando programação concorrente em C com PThreads. O sistema simula operações de pouso, desembarque e decolagem para voos domésticos e internacionais, gerenciando recursos limitados do aeroporto e tratando problemas de sincronização como deadlock e starvation.

2. Especificação do Sistema

2.1. Recursos do Aeroporto

O aeroporto simulado possui os seguintes recursos limitados:

- **3 Pistas:** Recursos exclusivos para pouso e decolagem
- **5 Portões:** Recursos exclusivos para embarque/desembarque
- **1 Torre de Controle:** Recurso compartilhado (máximo 2 operações simultâneas)

2.2. Operações por Tipo de Voo

Tipo	Operação	Ordem de Solicitação
Internacional	Pouso	Pista → Torre
	Desembarque	Portão → Torre
	Decolagem	Portão → Pista → Torre
Doméstico	Pouso	Torre → Pista
	Desembarque	Torre → Portão
	Decolagem	Torre → Portão → Pista

2.3. Sistema de Prioridades

- Voos internacionais possuem prioridade sobre domésticos
- Mecanismo anti-starvation implementado:
  - Estado crítico após 60s de espera
  - Crash simulado após 90s de espera

3. Arquitetura da Solução

3.1. Estruturas de Dados Principais

```c
typedef struct {
    int id;
    pthread_t thread_id;
    FlightType type;
    PlaneState state;
    time_t created_at;
    time_t waiting_since;
    time_t finished_at;
    bool is_in_critical_state;
} Plane;

typedef struct {
    sem_t tracks;
    sem_t gates;
    sem_t tower;
    pthread_mutex_t mutex_priority;
    int waiting_international_flights;
} Airport;
```

3.2. Estados do Sistema

- `WAITING_FOR_LANDING`: Aguardando recursos para pouso
- `DURING_LANDING`: Executando operação de pouso
- `WAITING_FOR_GATE`: Aguardando portão para desembarque
- `DURING_DISEMBARK`: Executando desembarque
- `WAITING_FOR_TAKEOFF`: Aguardando recursos para decolagem
- `DURING_TAKEOFF`: Executando decolagem
- `FINISHED`: Operações concluídas com sucesso
- `CRASHED_STARVATION`: Falha por starvation
- `CRASHED_DEADLOCK`: Falha por deadlock

4. Implementação de Sincronização

4.1. Prevenção de Deadlock

- Detecção de potencial deadlock baseada em timeout (30s)
- Diferentes ordens de aquisição de recursos entre tipos de voo
- Liberação ordenada de recursos

4.2. Prevenção de Starvation

- Sistema de prioridade com mutex específico
- Monitoramento de tempo de espera
- Estados críticos e crashes por timeout

4.3. Concorrência

- Semáforos para recursos limitados (pistas, portões, torre)
- Mutexes para proteção de seções críticas
- Thread pool para aviões
- Thread dedicada para geração contínua de aviões

5. Configuração e Parametrização

O sistema permite configuração flexível através dos arquivos `config.h` e `params.h`:

5.1. Parâmetros Configuráveis

- Duração da simulação (300s padrão)
- Número de pistas (3 padrão)
- Número de portões (5 padrão)
- Capacidade da torre (2 operações simultâneas)
- Tempos de timeout para starvation (60s crítico, 90s crash)

5.2. Perfis de Aeroportos

O sistema inclui 5 perfis de aeroportos reais com diferentes proporções de tráfego internacional:

Aeroporto	**Localização**	**% Internacional**
JFK	New York, USA	56%
Heathrow	London, UK	95%
Dubai	Dubai, UAE	99%
Atlanta	Atlanta, USA	15%
Guarulhos	São Paulo, Brazil	35%

6. Análise de Desempenho

6.1. Métricas Coletadas

- Total de aviões processados
- Taxa de sucesso
- Casos de starvation detectados
- Deadlocks identificados
- Máximo de aviões simultâneos
- Distribuição por estado final

6.2. Sistema de Logging

- Registro timestampado de todas as operações
- Rastreamento de estado por avião
- Identificação de problemas em tempo real
- Relatório final completo

7. Pontos Críticos da Implementação

7.1. Gestão de Recursos

A diferença na ordem de aquisição de recursos entre voos internacionais e domésticos é a principal fonte de complexidade, exigindo cuidado especial na implementação para evitar deadlocks.

7.2. Sistema de Prioridades

A implementação do sistema de prioridades para voos internacionais requer sincronização adicional através de `mutex_priority` para evitar starvation de voos domésticos.

7.3. Controle de Concorrência

O uso de múltiplos mutexes e semáforos requer ordem cuidadosa de aquisição e liberação para manter a consistência do sistema.

8. Resultados e Validação

8.1. Funcionalidades Implementadas

✓ Simulação completa do ciclo de voo (pouso, desembarque, decolagem)
✓ Diferentes estratégias por tipo de voo
✓ Sistema de prioridades funcionais
✓ Detecção de deadlock e starvation
✓ Configuração parameterizável
✓ Logging detalhado e relatórios
✓ Controle de parada limpa da simulação

8.2. Tratamento de Problemas

- **Deadlock:** Detecção por timeout e recuperação
- **Starvation:** Monitoramento ativo com estados de alerta
- **Race Conditions:** Proteção através de mutexes apropriados
- **Resource Leaks:** Cleanup adequado de recursos

9. Conclusões

O sistema Margolis demonstra uma implementação robusta de controle de tráfego aéreo com programação concorrente. A solução aborda efetivamente os desafios de sincronização em sistemas

de recursos limitados, implementando mecanismos adequados de prevenção de deadlock e starvation.

A arquitetura modular e configurável permite experimentação com diferentes cenários e cargas de trabalho, tornando o sistema adequado tanto para fins educacionais quanto para análise de desempenho de sistemas concorrentes.

9.1. Contribuições Principais

1. Implementação de diferentes estratégias de aquisição de recursos
2. Sistema robusto de detecção e prevenção de problemas de sincronização
3. Framework configurável para simulação de diferentes perfis de aeroportos
4. Logging abrangente para análise de comportamento do sistema

9.2. Trabalhos Futuros

- Implementação de algoritmos de scheduling mais sofisticados
- Análise de performance com diferentes configurações de recursos
- Extensão para múltiplos aeroportos conectados
- Interface gráfica para visualização em tempo real