

**NAMA : MUHAMMAD TAUFIK DIKRI**  
**NIM : 247200010**  
**PRODI : TEKNIK INFORMATIKA**

## **A. Pendahuluan**

Sistem operasi memiliki peran penting dalam mengelola sumber daya komputer seperti prosesor, memori, dan perangkat input/output. Dalam lingkungan multiprogramming, beberapa proses dapat berjalan secara bersamaan dan saling bersaing untuk mendapatkan sumber daya. Kondisi ini berpotensi menimbulkan deadlock, yaitu keadaan ketika dua atau lebih proses saling menunggu sumber daya sehingga tidak ada satu pun yang dapat melanjutkan eksekusi.

Untuk mengatasi permasalahan deadlock, sistem operasi menerapkan beberapa pendekatan, salah satunya adalah deadlock avoidance. Algoritma yang terkenal dalam pendekatan ini adalah Algoritma Banker's, yang bertujuan untuk memastikan sistem selalu berada dalam kondisi aman sebelum memberikan alokasi sumber daya kepada proses.

## **B. Konsep Algoritma Banker's**

Algoritma Banker's diperkenalkan oleh Edsger W. Dijkstra dan menggunakan analogi sistem perbankan. Dalam analogi tersebut, bank hanya akan memberikan pinjaman kepada nasabah jika bank yakin bahwa seluruh nasabah masih dapat menyelesaikan pinjamannya tanpa menyebabkan kebangkrutan.

Dalam sistem operasi, algoritma Banker's bekerja dengan cara meminta setiap proses untuk mendeklarasikan kebutuhan maksimum sumber daya yang mungkin dibutuhkan selama eksekusi. Berdasarkan informasi ini, sistem akan mengevaluasi setiap permintaan sumber daya dengan mensimulasikan alokasi tersebut. Jika hasil simulasi menunjukkan bahwa sistem tetap berada dalam kondisi aman (*safe state*), maka permintaan akan disetujui.

Komponen utama dalam algoritma Banker's meliputi:

- Available, yaitu jumlah sumber daya yang tersedia di sistem.
- Max, yaitu kebutuhan maksimum sumber daya dari setiap proses.
- Allocation, yaitu sumber daya yang telah dialokasikan ke proses.
- Need, yaitu sisa kebutuhan sumber daya yang dihitung dari selisih Max dan Allocation.

Sistem dikatakan berada dalam kondisi aman apabila terdapat setidaknya satu urutan eksekusi proses (*safe sequence*) yang memungkinkan seluruh proses selesai tanpa terjadi deadlock.

### **C. Mekanisme Kerja Algoritma Banker's**

Algoritma Banker's terdiri dari dua prosedur utama, yaitu Safety Algorithm dan Resource Request Algorithm.

Safety Algorithm digunakan untuk menentukan apakah kondisi sistem saat ini aman. Algoritma ini akan mencoba mengeksekusi proses secara berurutan dengan sumber daya yang tersedia. Jika suatu proses dapat dijalankan, maka sumber daya yang dimilikinya akan dikembalikan ke sistem setelah proses selesai. Proses ini diulang hingga seluruh proses selesai atau tidak ada proses yang dapat dijalankan.

Resource Request Algorithm digunakan ketika suatu proses mengajukan permintaan tambahan sumber daya. Permintaan tersebut tidak langsung diberikan, melainkan disimulasikan terlebih dahulu. Jika hasil simulasi menunjukkan sistem tetap aman, maka permintaan disetujui. Jika tidak, permintaan ditunda untuk mencegah deadlock.

### **D. Contoh Kasus**

Sebagai contoh, terdapat beberapa proses yang berjalan secara bersamaan dan membutuhkan beberapa jenis sumber daya. Sistem memiliki data mengenai alokasi sumber daya saat ini, kebutuhan maksimum setiap proses, serta jumlah sumber daya yang tersedia. Dengan menggunakan algoritma Banker's, sistem menghitung sisa kebutuhan setiap proses dan menentukan apakah permintaan sumber daya dapat dipenuhi tanpa membahayakan kondisi sistem.

Jika algoritma berhasil menemukan urutan eksekusi proses yang aman, maka sistem berada dalam kondisi aman. Namun, jika tidak ditemukan urutan tersebut, maka sistem berada dalam kondisi tidak aman dan permintaan sumber daya akan ditolak atau ditunda.

### **E. Implementasi Algoritma Banker's**

Algoritma Banker's dapat diimplementasikan menggunakan berbagai bahasa pemrograman seperti C, Java, atau Python. Umumnya, implementasi dimulai dengan membaca jumlah proses, jumlah jenis sumber daya, matriks Allocation, matriks Max,

serta vektor Available. Program kemudian menghitung matriks Need dan menjalankan Safety Algorithm untuk menentukan kondisi sistem.

Dalam praktiknya, algoritma Banker's lebih sering digunakan sebagai simulasi atau bahan pembelajaran. Hal ini disebabkan oleh keterbatasan algoritma yang mengharuskan sistem mengetahui kebutuhan maksimum setiap proses sejak awal, sesuatu yang sulit diterapkan pada sistem operasi modern.

#### **F. Kelebihan dan Kekurangan**

Kelebihan utama algoritma Banker's adalah kemampuannya dalam mencegah deadlock sebelum terjadi. Algoritma ini memberikan pendekatan yang sistematis dan terstruktur dalam pengelolaan sumber daya.

Namun, algoritma Banker's juga memiliki beberapa kekurangan, seperti kebutuhan informasi maksimum sumber daya yang tidak selalu realistis, serta overhead komputasi yang cukup besar jika jumlah proses dan sumber daya meningkat.

#### **G. Kesimpulan**

Algoritma Banker's merupakan algoritma penting dalam pembahasan sistem operasi, khususnya pada topik deadlock avoidance. Dengan memastikan sistem selalu berada dalam kondisi aman sebelum mengalokasikan sumber daya, algoritma ini mampu mencegah terjadinya deadlock.

Meskipun penerapannya di sistem operasi modern terbatas, algoritma Banker's tetap memiliki nilai edukatif yang tinggi dan membantu dalam memahami konsep manajemen sumber daya dan deadlock secara mendalam.