16 System Software 系统软件

16.1 Purposes of an Operating System 操作系统的目的

Multi-tasking 多任务

Managing the execution of many programs that appear to run at the same time.

process 进程

a program that has started to be executed.

scheduling 调度

managing the processes running on the CPU. The algorithm can be preemptive or non-preemptive.

thread 线程

part of a program, handled as an individual process when being executed

process states 进程状态

the states of a process requiring execution the scheduler refers to when making decisions. At any given moment, a process can be either ready (waiting for CPU time), running (currently using the CPU), or blocked (neither using nor waiting to use the CPU, such as waiting for an event).

burst time 突发时间

the time a process needs to spend running on the CPU.

preemptive scheduling 抢占式调度

types of scheduling that can halt a process that would otherwise continue running undisturbed, switching it from running state to ready state. Example algorithms include SRTF and round robin.

quantum 时间片

the unit of CPU time allocation in preemptive scheduling. Each process controls the CPU for one quantum, after which the scheduler may decide to let another process take control of the CPU, even if the current one hasn't finished its burst time.

non-preemptive scheduling 非抢占式调度

types of scheduling under which a running process can keep running until it switches to blocked state. Example algorithms include FCFS and SJF.

first come first served scheduling (FCFS) 先到先得式调度

a type of non-preemptive scheduling. Process in waiting state are put in a queue, and every time the CPU becomes available, the process at the front of the queue is selected to be run next.

shortest job first scheduling (SJF) 最短任务优先式调度

a type of non-preemptive scheduling. Every time the CPU become available, the process with the shortest burst time is selected to be run next.

shortest remaining time first scheduling (SRTF) 最短剩余时间优先式调度

a type of preemptive scheduling. Every time the CPU become available, the process with the shortest remaining time is selected to be run next.

round-robin scheduling (RR) 轮转式调度

a type of preemptive scheduling. Once a process has spent its quantum, it is removed from the CPU and place at the rear of the ready queue.

virtual memory 虚拟内存

using secondary storage to temporarily simulate additional main memory, so the CPU appears to be able to access more memory space than the available RAM. The data not currently in use are swapped from the main memory (RAM) to the virtual memory.

Paging 内存分页

form of memory management which divides up memory into same-size blocks called pages. In virtual memory management where data is read from / written to secondary storage, page is the smallest unit.

Segmentation 内存分段

form of memory management which breaks the memory into variable-size blocks called segments.

Interrupt handling 中断处理

transferring control to another routine when a service is required.

16.2 Translation Software 程序语言处理软件

front-end analysis 前端分析

the first part of compilation, aiming at establishing the meaning of the source code. It consists of lexical analysis, syntax analysis. semantic analysis and intermediate code generation, in that order.

back-end synthesis 后端合成

the second part of compilation that generates the object (machine) code from the intermediate code. Optimisation is usually done before the machine code is produced.

lexical analysis 词法分析

the first stage of compilation. The comments and whitespaces are removed. Then the source code (as a sequence of characters) is identified via pattern-matching as a sequence of lexemes, each of which correspond to a token.

symbol table 符号表

a data structure in which each record contains the name and attribute of an identifier. It is generated during the lexical analysis.

syntax analysis 语法分析

the second stage of compilation, also known as **parsing**. Firstly, the list of tokens is checked for syntax error. If there is none, a parsing algorithms is used to interpret token list and produce a syntax tree (parse tree).

semantic analysis 语义分析

the third stage of compilation that is about establishing the full meaning of the code. It constructs an annotated abstract syntax tree.

intermediate code generation 中间代码生成

the fourth stage of compilation that converts the abstract syntax tree into an intermediate code.

optimisation 优化

a stage of compilation aiming at minimising a program’s execution time and memory requirement.

code generation 机器码生成

the last stage of compilation. converting the optimised intermediate code into an executable form.