Process migration

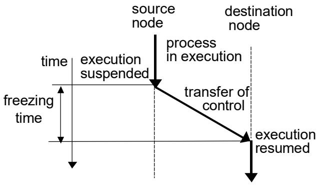
- why migrate processes
- main concepts
- PM design objectives
- design issues
- freezing and restarting a process
- address space transfer
- handling messages for moved processes
- handling co-processes

Advantages of process migration

- balancing the load:
  - reduces average response time of processes
  - speeds up individual jobs
  - gains higher throughput
- moving the process closer to the resources it is using:
  - utilizes resources effectively
  - reduces network traffic
- being able to move a copy of a process (replicate) on another node improves system reliability
- a process dealing with sensitive data may be moved to a secure machine (or just to a machine holding the data) to improve security

Process migration

- Load balancing (load sharing) policy determines:
  - if the process needs to be moved (**migrated**) from one node of the distributed system to another.
  - which process needs to be migrated
  - what is the node to which the process it to be moved
- process migration mechanism deals with the actual transfer of the process



Desirable features of good process migration mechanism

- Transparency
  - object access level - access to objects (such as files and devices) by process can be done in location -independent manner.
  - system call and interprocess communication level - the communicating processes should not notice if one of the parties is moved to another node, system calls should be equivalent
- Minimal interference (with process execution) - minimize freezing time
- Minimal residual dependencies - the migrated process should not depend on the node it migrated from or:
  - previous node is still loaded
  - what if the previous node fails?
- Efficiency:
  - minimize time required to migrate a process
  - minimize cost relocating the process
  - minimize cost of supporting the migrated process after migration

Parts of process migration mechanism

- freezing the process on its source node and restarting at destination node
- moving the process' address space
- forwarding messages meant for the migrant process
- handling communication between cooperating processes that are separated (handling co-processes)

Freezing and restarting of process

- blocks the execution of the migrant process, postponing all external communication
  - immediate blocking - when not executing system call
  - postponed blocking - when executing certain system calls
- wait for I/O operations:
  - wait for fast I/O - disk I/O
  - arrange to gracefully resume slow I/O operations at destination - terminal I/O, network communication
- takes a "snapshot" of the process state
  - relocatable information - register contents, program counter, etc.
  - open files information - names, identifiers, access modes, current positions of file pointers, etc.
- transfers the process state to the destination
- restarts the process on destination node

Address space transfer

- Process state (a few kilobytes):
  - contents of registers, program counter, I/O buffers, interrupt signals, etc.
- Address space (several megabytes) - dominates:
  - program's code, data and stack
- Several approaches to address space transfer
  - total freezing - no execution is done while address space is transferred - simplest, slowest
  - pretransferring - address space is transferred while the process is still running on the source node, after the transfer, the modified pages are picked up
  - transfer on reference - the process is restarted before the address space is migrated - the pages are fetched from the source node as the process needs them

Message forwarding

- Three types of messages:
  1. received when the process execution is stopped on the source node and has not restarted on the destination node
  2. received on the source node after the execution started on destination node
  3. sent to the migrant process after it started execution on destination node
- approaches:
  - re-sending - messages of type 1 and 2 are either dropped or negatively ack-ed, the sender is notified and it needs to locate the migrant process - nontransparent
  - origin site - origin node keeps the info on the current location of the process created there, all messages are sent to origin which forwards them to migrant process - expensive, not fault tolerant

Message forwarding (cont.)

- Approaches to message forwarding:
  - link traversal
    - messages of type 1 are queued and sent to destination node as part of migration procedure
    - forwarding address (link) is left on source node to redirect messages of type 2 and 3, link contains the system-wide unique id of a process and its last known location - may not be efficient or fault tolerant
  - link update - during the transfer the source node sends the notification (link update) of the transfer to all the nodes to which the process communicates:
    - messages of type 1 and 2 are forwarded by the source node
    - messages of type 3 are sent directly to the destination node

Co-processes handling

- Need to provide efficient communication between a parent process and subprocesses
- no separation of co-processes:
  - disallow migration of a process if it has children
  - migrate children together with process
    - logical host concept - co-processes are always executed on one logical host, and logical host is migrated atomically
- home node (origin site):
  - all communication between co-processes is handled through home node - expenisve