Virtual Memory
During any interval of time a process favours a subset of its pages.
During any interval of time a process favours a subset of its pages.

The memory reference patterns show a high correlation between recent past and immediate future.
During any interval of time a process favours a subset of its pages.

The memory reference patterns show a high correlation between recent past and immediate future.

The frequency at which a page is referenced is a slowly changing function of time.
During any interval of time a process favours a subset of its pages.

The memory reference patterns show a high correlation between recent past and immediate future.

The frequency at which a page is referenced is a slowly changing function of time.

The working set of a process $W(t, \theta)$ is defined to be the set of pages referenced in the last $\theta$ memory references.
During any interval of time a process favours a subset of its pages.
The memory reference patterns show a high correlation between recent past and immediate future.
The frequency at which a page is referenced is a slowly changing function of time.
The working set of a process $W(t, \theta)$ is defined to be the set of pages referenced in the last $\theta$ memory references.
$W(t, \theta) = \{ i \in N | \text{page } i \text{ appears among } r_{t-\theta+1}, \ldots, r_t \}$
A program should be run if and only if its working set is in memory.
Working set principle

- A program should be run if and only if its working set is in memory.
- A page may not be removed if it is within the working set of a process.
A page should be replaced only if it is not a member of the working set.
A page should be replaced only if it is not a member of the working set. Otherwise, the page fault rate is likely to be high, since pages with high probability of being accessed in the near future are replaced.
A page should be replaced only if it is not a member of the working set.

Otherwise, the page fault rate is likely to be high, since pages with high probability of being accessed in the near future are replaced.

The number of page frames allocated to any process corresponds to the size of its working set. The size of the working set is the number of distinct pages contained in it.
A page should be replaced only if it is not a member of the working set. Otherwise, the page fault rate is likely to be high, since pages with high probability of being accessed in the near future are replaced. The number of page frames allocated to any process corresponds to the size of its working set. The size of the working set is the number of distinct pages contained in it. If new page frames are necessary for some processes and the pool of free pages is depleted, the operating system swaps some processes out to disc (secondary storage).
A process whose entire working set is not contained in memory should not be run. It should be suspended and swapped out to secondary storage.
A process whose entire working set is not contained in memory should not be run. It should be suspended and swapped out to secondary storage.

Conversely, when a reasonable number of free page frames become available in the free pool, the operating system will activate more processes.
A process whose entire working set is not contained in memory should not be run. It should be suspended and swapped out to secondary storage.

Conversely, when a reasonable number of free page frames become available in the free pool, the operating system will activate more processes.

The working set principle provides for implementation of a local replacement and allocation policy that helps prevent thrashing, while increasing the degree of multiprogramming.
The parameter $\theta$ is very important in determining the performance of an implementation of the working set principle.
• The parameter $\theta$ is very important in determining the performance of an implementation of the working set principle.

• The A very small value of $\theta$ results in the information being too localised. It does not adequately define the set of pages necessary to be in memory and leads to an increased frequency of page faults.
The parameter \( \theta \) is very important in determining the performance of an implementation of the working set principle. A very small value of \( \theta \) results in the information being too localised. It does not adequately define the set of pages necessary to be in memory and leads to an increased frequency of page faults.

A very high value results in multiple localities, and leads to over allocation of page frames.
The parameter $\theta$ is very important in determining the performance of an implementation of the working set principle.

The very small value of $\theta$ results in the information being too localised. It does not adequately define the set of pages necessary to be in memory and leads to an increased frequency of page faults.

A very high value results in multiple localities, and leads to over allocation of page frames.

Accurate tracking of working sets requires updates after every memory reference. This is very expensive, and most systems use approximations, by making updates after every 1000 memory references, or some similar high value.
The parameter $\theta$ is very important in determining the performance of an implementation of the working set principle. A very small value of $\theta$ results in the information being too localised. It does not adequately define the set of pages necessary to be in memory and leads to an increased frequency of page faults.

A very high value results in multiple localities, and leads to over allocation of page frames.

Accurate tracking of working sets requires updates after every memory reference. This is very expensive, and most systems use approximations, by making updates after every 1000 memory references, or some similar high value.

The result of this periodic tracking can be stored in *referenced bits*. This can be in the PMT or in some bit array stored elsewhere. The referenced bits of resident processes are cleared or set at the time of checking periodically.
Hardware support for virtual memory

- Instruction interruptibility and restartability.
Hardware support for virtual memory

- Instruction interruptibility and restartability.
- A collection of page status bits with each page descriptor.
Hardware support for virtual memory

- Instruction interruptibility and restartability.
- A collection of page status bits with each page descriptor.
- A TLB to accelerate address translations.
• Presence bit is used to detect items missing from main memory.
- Presence bit is used to detect items missing from main memory.
- Written-into/modified bit keeps track of the pages which have been modified.
- Presence bit is used to detect items missing from main memory.
- Written-into/modified bit keeps track of the pages which have been modified.
- Referenced bit keeps track of whether the page was referenced recently or not.
It is possible to implement virtual memory with segmentation instead of paging.
Virtual memory management with segmentation

- It is possible to implement virtual memory with segmentation instead of paging.
- These implementations inherit the benefits of sharing and protection associated with segment based memory management.
It is possible to implement virtual memory with segmentation instead of paging.

These implementations inherit the benefits of sharing and protection associated with segment based memory management.

Placement policies are aided explicit awareness of the types of informations in particular segments.
It is possible to implement virtual memory with segmentation instead of paging.

These implementations inherit the benefits of sharing and protection associated with segment based memory management.

Placement policies are aided explicit awareness of the types of informations in particular segments.

A “working set” should consist of at least one segment of each of code, data and stack.
It is possible to implement virtual memory with segmentation instead of paging.

These implementations inherit the benefits of sharing and protection associated with segment based memory management.

Placement policies are aided explicit awareness of the types of informations in particular segments.

A “working set” should consist of at least one segment of each of code, data and stack.

However contiguous placement requirement complicate management of both main memory and secondary storage.
Paging is convenient for memory management, but logical issues are less easy to deal with.
Paging is convenient for memory management, but logical issues are less easy to deal with.

In paging issues of page replacement etc. can only be dealt with on the basis of probabilistic algorithms and observed process behaviour by the system.
Paging is convenient for memory management, but logical issues are less easy to deal with.

In paging issues of page replacement etc. can only be dealt with on the basis of probabilistic algorithms and observed process behaviour by the system.

In segmentation, logical program information can be used for these decisions.
Paging is convenient for memory management, but logical issues are less easy to deal with.

In paging issues of page replacement etc. can only be dealt with on the basis of probabilistic algorithms and observed process behaviour by the system.

In segmentation, logical program information can be used for these decisions.

Virtual memory systems exist with a mix of paging and segmentation.
- Paging is convenient for memory management, but logical issues are less easy to deal with.
- In paging issues of page replacement etc. can only be dealt with on the basis of probabilistic algorithms and observed process behaviour by the system.
- In segmentation, logical program information can be used for these decisions.
- Virtual memory systems exist with a mix of paging and segmentation.
- In such systems, the address translation process is highly complicated, and involves often several levels of depth to reach an actual physical address.