

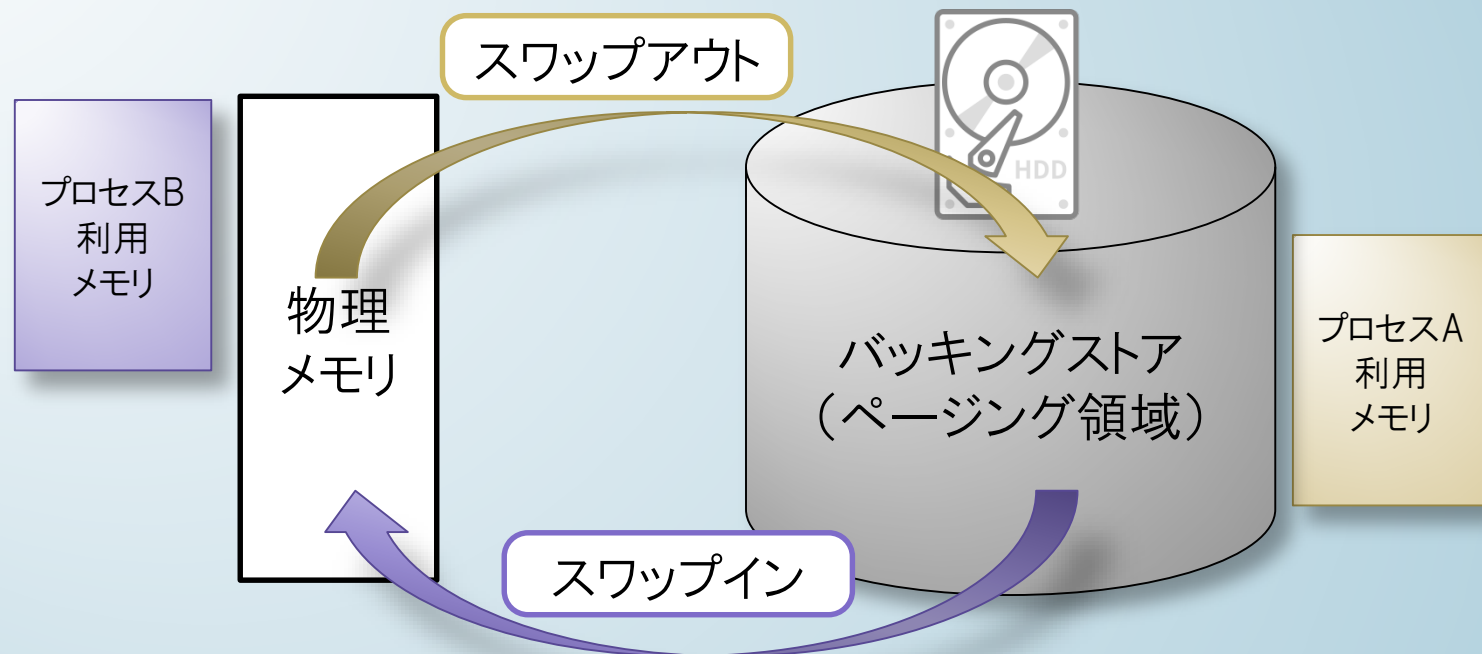
オペレーティングシステム

第7回 階層記憶

情報科学メジャー
鎬木崇史

プロセス・スワッピング

実行中のプロセスに必要なデータはすべてメモリ上にはないと実行できない



- 情報の交換単位が大きいため、入出力に時間がかかる
- 他のプロセスにCPUを譲らないと処理スループットが低下する
- 並行動作するプロセスのメモリ量の合計は物理メモリ量を超えることができるが、プロセス単体の利用できるメモリ量は物理メモリ量によって上限が決まる

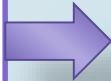


デマンドページング

ページテーブル

論理	物理
0	8
1	3
2	4
3	
4	

新たにmmap()で
論理ページを
割り当てる



mmap()で割り当てた論理ページは
最初にアクセスするまでは
物理ページを確保しない

物理 ページ番号
0
1
2
3
4
5
6
7
8
9
10
...



デマンドページング

ページテーブル

論理	物理
0	8
1	3
2	4
3	10
4	

はじめてアクセス



マイナー
ページフォルト
発生

mmap()で割り当てた論理ページは
最初にアクセスするまでは
物理ページを確保しない

アクセス要求があった論理ページに
物理ページを割り当てる
→演習1

物理 ページ番号
0
1
2
3
4
5
6
7
8
9
10
...



ページアウト

ページテーブルA

論理	物理
0	8
1	3
2	4
3	*
4	

新たにmmap()で
論理ページを
割り当てて
アクセスするが
物理ページに
空きがない！

ページテーブルB

論理	物理
0	0
1	1
2	5
3	
4	

物理 ページ番号
0
1
2
3
4
5
6
7
8
9
10
...



ページアウト

ページテーブルA

論理	物理
0	8
1	3
2	4
3	5
4	

無事にページが
割り当てられる

ページテーブルB

論理	物理
0	0
1	1
2	B0
3	
4	

使用頻度の低い
ページに着目

ページ
書き込み

物理 ページ番号
0
1
2
3
4
5
6
7
8
9
10
...

バッキング ストア
0
1
2
...
...
...
...
...
...
...
...

バッキングストアに
退避させる

物理ページに
空きができる



ページイン

ページテーブルA

論理	物理
0	8
1	3
2	4
3	5
4	

ページテーブルB

論理	物理
0	0
1	1
2	2
3	

物理 ページ番号
0
1
2
3
4
5
6
7
8
9
10

バッキング ストア
0
1
2
3
4
5
6
7
8
9
10
...

物理ページに
空きを見つけて
バッキングストアから
データを移動する

無事にページが
割り当てられる

バッキングストア
にあるページに
アクセス

メジャー
ページフォルト
発生

ページテーブルを
書き換える



参照の局所性

時間的局所性

近い時間に同じ範囲にアクセスする可能性が高い

空間的局所性

アドレス空間で近い領域の近いアドレスが参照される可能性が高い

ワーキングセット

プロセスが一定時間に利用する仮想メモリの集合

ワーキングセットが物理ページに存在しないとスラッシングの危険性

ページフォルト率

論理ページあたりのページフォルト発生率

→これを用いてワーキングセットの大きさが適切かを近似する



置き換えアルゴリズム

先入れ先出しアルゴリズム (First in First out)
物理メモリ上の最も古いページを置き換える

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF															
物理 ページ数 =4															



置き換えアルゴリズム

先入れ先出しアルゴリズム(First in First out)
物理メモリ上の最も古いページを置き換える

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF	*	*	*	*	*	*	*	*	*				*	*	
物理 ページ数 =4	0	0	0	0	1	2	3	4	0	0	0	0	1	2	2
		1	1	1	2	3	4	0	1	1	1	1	2	5	5
			2	2	3	4	0	1	2	2	2	2	5	3	3
				3	4	0	1	2	5	5	5	5	3	4	4



置き換えアルゴリズム

先入れ先出しアルゴリズム (First in First out)

物理メモリ上の最も古いページを置き換える

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF															
物理 ページ数 =5															



置き換えアルゴリズム

先入れ先出しアルゴリズム(First in First out)

物理メモリ上の最も古いページを置き換える

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF	*	*	*	*	*				*	*	*	*	*	*	*
物理 ページ数 =5	0	0	0	0	0	0	0	0	1	2	3	4	5	0	1
		1	1	1	1	1	1	1	2	3	4	5	0	1	2
			2	2	2	2	2	2	3	4	5	0	1	2	3
				3	3	3	3	3	4	5	0	1	2	3	4
					4	4	4	4	5	0	1	2	3	4	5

物理ページが増えるとかえってページフォルトが増加している
→Belady's anomaly



置き換えアルゴリズム

最適アルゴリズム (Optimal)

将来最も長い期間参照されないページを置き換える
実装は不可能だが、他のアルゴリズムとの比較に用いる

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF															
物理 ページ数 =4															



置き換えアルゴリズム

最適アルゴリズム (Optimal)

将来最も長い期間参照されないページを置き換える
実装は不可能だが、他のアルゴリズムとの比較に用いる

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF	*	*	*	*	*				*				*	*	
物理 ページ数 =4	0	0	0	0	0	0	0	0	0	0	0	0	3	3	3
		1	1	1	1	1	1	1	1	1	1	1	1	4	4
			2	2	2	2	2	2	2	2	2	2	2	2	2
				3	4	4	4	4	5	5	5	5	5	5	5



置き換えアルゴリズム

最適アルゴリズム (Optimal)

将来最も長い期間参照されないページを置き換える
実装は不可能だが、他のアルゴリズムとの比較に用いる

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF															
物理 ページ数 =5															



置き換えアルゴリズム

最適アルゴリズム (Optimal)

将来最も長い期間参照されないページを置き換える
実装は不可能だが、他のアルゴリズムとの比較に用いる

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF	*	*	*	*	*				*					*	
物理 ページ数 =5	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4
		1	1	1	1	1	1	1	1	1	1	1	1	1	1
			2	2	2	2	2	2	2	2	2	2	2	2	2
				3	3	3	3	3	3	3	3	3	3	3	3
					4	4	4	4	5	5	5	5	5	5	5



置き換えアルゴリズム

最長未使用時間アルゴリズム (Least Recently Used)

最も長い間参照していないページを置き換える

→ 最終アクセス時間を記録する方法

ページ全体から探すオーバーヘッドが大きい

→ Last in First out (スタック) → オーバーヘッドが小さい

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF															
物理 ページ数 =4															



置き換えアルゴリズム

最長未使用時間アルゴリズム (Least Recently Used)

最も長い間参照していないページを置き換える

→ 最終アクセス時間を記録する方法

ページ全体から探すオーバーヘッドが大きい

→ Last in First out (スタック) → オーバーヘッドが小さい

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF	*	*	*	*	*	*	*	*	*				*	*	*
物理 ページ数 =4	0	0	0	0	1	2	3	4	0	1	2	5	0	1	2
		1	1	1	2	3	4	0	1	2	5	0	1	2	3
			2	2	3	4	0	1	2	5	0	1	2	3	4
				3	4	0	1	2	5	0	1	2	3	4	5



置き換えアルゴリズム

最長未使用時間アルゴリズム (Least Recently Used)

最も長い間参照していないページを置き換える

→ 最終アクセス時間を記録する方法

ページ全体から探すオーバーヘッドが大きい

→ Last in First out (スタック) → オーバーヘッドが大きい

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF															
物理 ページ数 =5															



置き換えアルゴリズム

最長未使用時間アルゴリズム (Least Recently Used)

最も長い間参照していないページを置き換える

→ 最終アクセス時間を記録する方法

ページ全体から探すオーバーヘッドが大きい

→ Last in First out (スタック) → オーバーヘッドが大きい

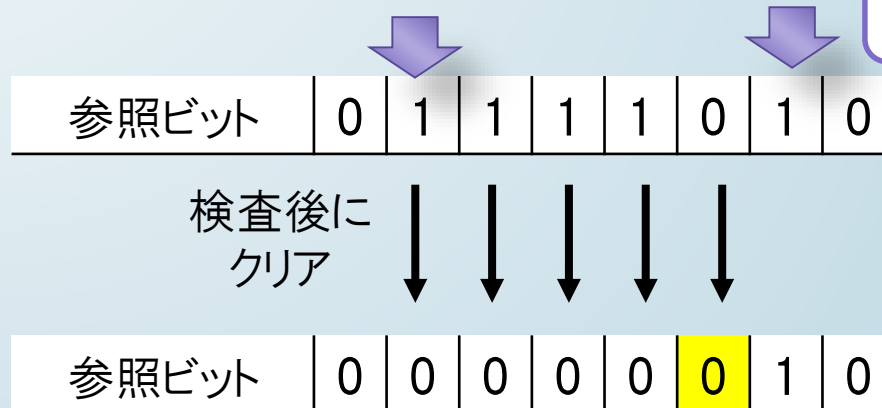
参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
PF	*	*	*	*	*				*				*	*	*
物理 ページ数 =5	0	0	0	0	0	1	2	3	4	4	4	4	5	0	1
		1	1	1	1	2	3	4	0	1	2	5	0	1	2
			2	2	2	3	4	0	1	2	5	0	1	2	3
				3	3	4	0	1	2	5	0	1	2	3	4
					4	0	1	2	5	0	1	2	3	4	5



置き換えアルゴリズム

クロックアルゴリズム

最終アクセス時間やスタックを用いずに参照ビットを用いる手法

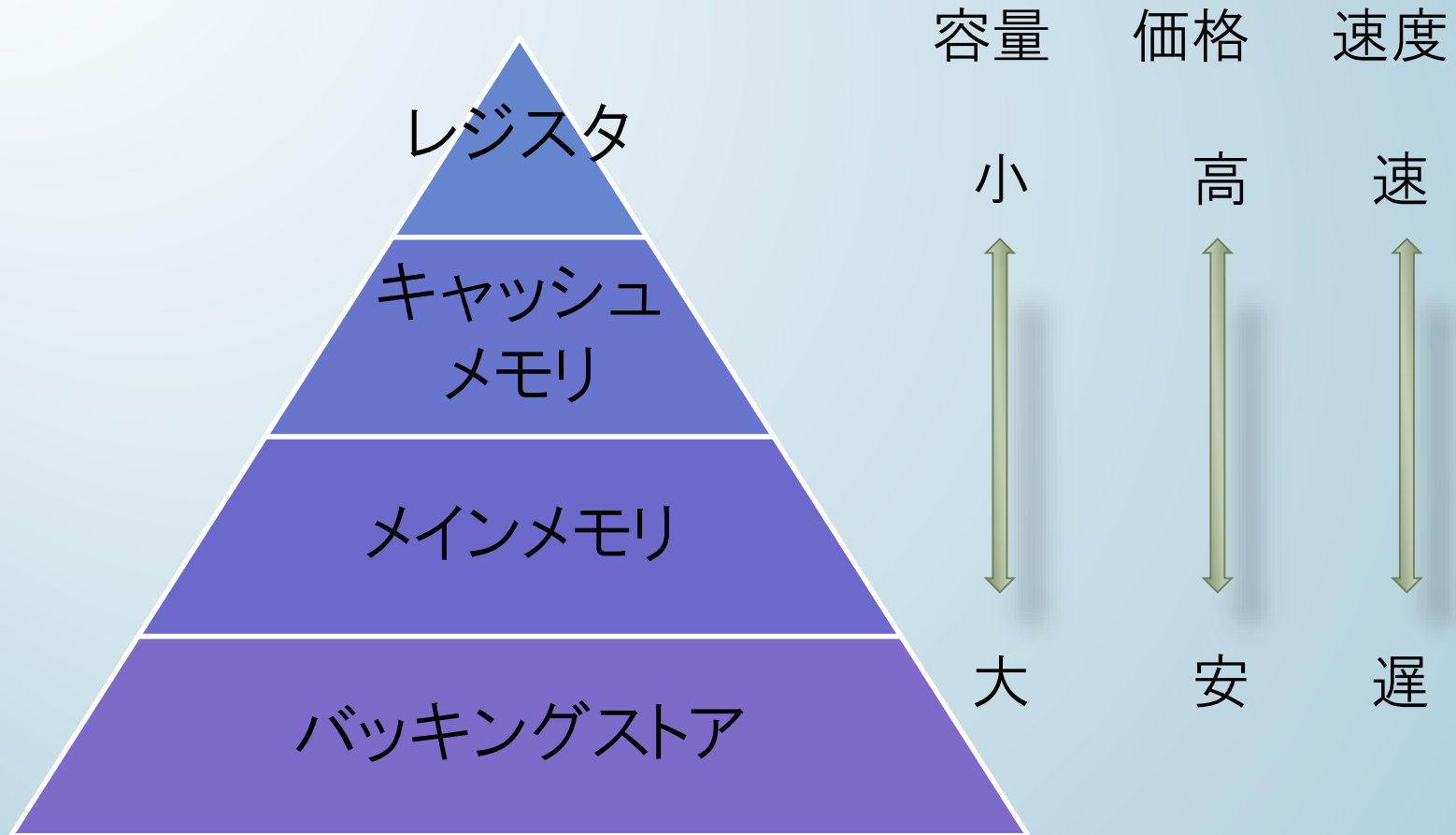


次はここからスタート

しばらく使っていないページ！
置き換え対象として選択



記憶装置の階層構造



キャッシュの置き換えもページングの置き換えアルゴリズムと同様に実装されている 演習2, 3



課題1

Moodleの demand-paging.c は

- 1.メモリ獲得前でEnterキーの入力を待つ
- 2.メモリ100MBを獲得し、再びEnterキーの入力を待つ
3. 獲得したメモリの最初から最後まで1ページずつアクセスし、10MBごとに報告する
- 4.100MBに達するとその旨を表示して終了する

```
$ cc -o page-demand page-demand.c  
$ ./page-demand
```

```
student@testvm:~$ ./demand-paging  
Thu May 16 03:07:37 2019: before allocation, please press Enter key  
  
Thu May 16 03:07:55 2019: allocated 100MB, please press Enter key  
  
Thu May 16 03:08:02 2019: touched 10MB  
Thu May 16 03:08:03 2019: touched 20MB  
Thu May 16 03:08:04 2019: touched 30MB  
Thu May 16 03:08:05 2019: touched 40MB  
Thu May 16 03:08:06 2019: touched 50MB  
Thu May 16 03:08:07 2019: touched 60MB  
Thu May 16 03:08:08 2019: touched 70MB  
Thu May 16 03:08:09 2019: touched 80MB  
Thu May 16 03:08:10 2019: touched 90MB  
Thu May 16 03:08:11 2019: touched 100MB, please press Enter key
```



課題1 つづき

一方、vsz-rss.shは仮想メモリサイズ、物理メモリサイズ、メジャーPF、マイナーPFを一定時間ごとに報告するプログラムである。

```
student@testvm:~$ ./vsz-rss.sh
```

2019年	5月	16日	木曜日	03:07:42	JST:	3850	demand-paging	4508	744	0	78
2019年	5月	16日	木曜日	03:07:43	JST:	3850	demand-paging	4508	744	0	78
2019年	5月	16日	木曜日	03:07:44	JST:	3850	demand-paging	4508	744	0	78
2019年	5月	16日	木曜日	03:07:45	JST:	3850	demand-paging	4508	744	0	78
2019年	5月	16日	木曜日	03:07:46	JST:	3850	demand-paging	4508	744	0	78
2019年	5月	16日	木曜日	03:07:47	JST:	3850	demand-paging	4508	744	0	78
2019年	5月	16日	木曜日	03:07:48	JST:	3850	demand-paging	4508	744	0	78
2019年	5月	16日	木曜日	03:07:49	JST:	3850	demand-paging	4508	744	0	78
2019年	5月	16日	木曜日	03:07:50	JST:	3850	demand-paging	4508	744	0	78
2019年	5月	16日	木曜日	03:07:51	JST:	3850	demand-paging	4508	744	0	78

左から

仮想メモリサイズ、物理メモリサイズ、
メジャーPF、マイナーPF

デマンドページングにおいて、以下の実験を通して

- 1.仮想メモリ確保のタイミング
 - 2.物理メモリ確保のタイミング
 - 3.ページフォルトの発生のタイミング(メジャーページフォルト(PF)の有無)
- について報告せよ。

```
$ cc -o demand-paging demand-paging.c  
$ ./demand-paging
```

```
$ chmod +x vsz-rss.sh  
$ ./vsz-rss.sh
```



課題2

システム内にあるキャッシュについて、どのようなものがあるかまとめて報告せよ。

```
$ cat /sys/devices/system/cpu/cpu0/cache/index<N>/<DATATYPE>
```

ただし

index<N> → キャッシュの数だけ存在する 例) index0, index1, index2, index3 ...

<DATATYPE>は level, type, size など

level → キャッシュの階層 若い数ほどレジスタに近い

type → Dataならデータのみ、Codeならプログラムコードのみ、Unifiedなら両方

size → キャッシュの容量

ファイル名	level	type	size
index0	1	Data	32KB



課題3

Moodleの cache.c は(1)コマンドの引数で与えた容量のメモリを獲得し(2)獲得したメモリ内にシーケンシャル(連続して)アクセスしたときの所要時間を表示するプログラムである。

引数に与える容量を4KB～32MBへと変化させたとき、アクセス速度はどのように変化したかキャッシュの観点から考察せよ。

```
$ cc -o cache cache.c  
$ for i in 4 8 16 32 64 128 256 512 1024 2048 4096 8192 16384 32768 ; (続く)  
do ./cache $i ; done
```

