Gruppe: 15 Luis Nothvogel, Tobias Schoch

18.1

Bei einem größeren Arbeitsspeicher brauchen wir logischerweise auch eine größere Page Table Größe, wenn weiterhin gleichviele Pages in dem Arbeitsspeicher Platz finden sollen.
Wenn also der virtuelle Arbeitsspeicher um 20% größer wird, dann muss auch jede Page Table 20% größer werden, dass weiterhin gleichviele Pages im Arbeitsspeicher sich befinden. Wenn sich die Page Table Größe verdoppelt, dann ist allerdings für lediglich die Hälfte an Pages Platz im jeweiligen virtuellen Arbeitsspeicher.

Wenn der physikalische Arbeitsspeicher sich hingegen vergrößert, bleibt die Page Table Größe hingegen gleich, da diese sich lediglich an der Größe des virtuellen Arbeitsspeichers orientiert. Allerdings steigt dadurch die Anzahl der vorhandenen Page Tables.

Es ist natürlich schlecht zu große Page Tables zu haben, da hierdurch sehr viel Arbeitsspeicher verschwendet wird, da die meisten Prozesse nicht übermäßig viel Arbeitsspeicher verwenden.

18.2

Je höher der Wert (0-100) für die Flag -u ist, desto höher ist die Wahrscheinlichkeit, dass eine Page reserviert ist.

```
Page Table (from entry 0 down to the max size)
  [       0]   0x00000000
  [       1]   0x00000000
  [       2]   0x00000000
  [       3]   0x00000000
  [       4]   0x00000000
  [       5]   0x00000000
  [       6]   0x00000000
  [       7]   0x00000000
  [       8]   0x00000000
  [       9]   0x00000000
  [      10]   0x00000000
  [      11]   0x00000000
  [      12]   0x00000000
  [      13]   0x00000000
  [      14]   0x00000000
  [      15]   0x00000000
```

Bei einer Wahrscheinlichkeit von 0% Page Reservierung, wird logischerweise auch kein Speicher reserviert.

```
Page Table (from entry 0 down to the max size)
  [       0]   0x80000018
  [       1]   0x00000000
  [       2]   0x00000000
  [       3]   0x00000000
  [       4]   0x00000000
  [       5]   0x80000009
  [       6]   0x00000000
  [       7]   0x00000000
  [       8]   0x80000010
  [       9]   0x00000000
  [      10]   0x80000013
  [      11]   0x00000000
  [      12]   0x8000001f
  [      13]   0x8000001c
  [      14]   0x00000000
  [      15]   0x00000000
```

Mit der Flag „-u 25“ werden bereits 6 von 16 Pages reserviert.

```
Page Table (from entry 0 down to the max size)
  [       0]   0x80000018
  [       1]   0x00000000
  [       2]   0x00000000
  [       3]   0x8000000c
  [       4]   0x80000009
  [       5]   0x00000000
  [       6]   0x8000001d
  [       7]   0x80000013
  [       8]   0x00000000
  [       9]   0x8000001f
  [      10]   0x8000001c
  [      11]   0x00000000
  [      12]   0x8000000f
  [      13]   0x00000000
  [      14]   0x00000000
  [      15]   0x80000008
```

Mit der Flag „-u 50“ werden schon 9 von 16 Pages reserviert.

```
Page Table (from entry 0 down to the max size)
  [       0]   0x80000018
  [       1]   0x80000008
  [       2]   0x8000000c
  [       3]   0x80000009
  [       4]   0x80000012
  [       5]   0x80000010
  [       6]   0x8000001f
  [       7]   0x8000001c
  [       8]   0x80000017
  [       9]   0x80000015
  [      10]   0x80000003
  [      11]   0x80000013
  [      12]   0x8000001e
  [      13]   0x8000001b
  [      14]   0x80000019
  [      15]   0x80000000
```

Mit der Flag „-u 75“ sind alle Pages reserviert, daher sind logischerweise auch bei „-u 100“ alle Pages reserviert.

Zusammenfassend: Desto höher die Wahrscheinlichkeit bei -u gesetzt wurde, desto wahrscheinlicher sind Pages reserviert.

Die Tutoren haben unsere Gruppe gebeten eine Adresse umzuwandeln, dass es ersichtlich ist, dass wir es auch verstanden haben.

Wir wandeln die VP[6] aus dem letzten Screenshot um: **0x8000001f**
Die 8 am Anfang ist das Valid Bit, ob eine Page gültig ist. Unsere Adresse ist gültig, da die 8 vorhanden ist. Der Rest der Adresse wir von einer Hexadezimal- in eine Binärzahl umgewandelt.

Dabei kommt die **11111** raus. Im Anschluss nehmen wir den 2er Logarithmus des virtuellen Adressraumes. Dieser ist **14** (2^14). Das bedeutet, dass unsere Translation 14 Stellen hat. Nun nehmen wir die Page Größe als 2er Logarithmus. Dabei kommt **10** heraus (2^10). Das heißt der Offset ist 10 Stellen lang und die ersten 4 Stellen beschreiben die Virtual Page Number.

Wenn wir nun eine Page Table hätten, könnten wir aus dieser die „Physical Frame Number“ auslesen mit unserer „Virtual Page Number“ (0000). Dabei ersetzt man im Anschluss die „Virtual Page Number“ mit der „Physical Frame Number“. Der Offset bleibt bei der Translation gleich.

Virtual Page Number — Offset

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**

18.3

Unter anderem die **dritte Parameter Kombination**. Da lediglich 2 virtuelle Adressbereiche in den physischen Arbeitsspeicher passen, was viel zu wenig ist.

512 MB physischer Arbeitsspeicher / 256 MB virtueller Arbeitsspeicher = 2

Im Vergleich hat er eine sehr kleine Page Größe, was dazu führt, dass sehr viele Pages in den virtuellen Adressspeicher passen im Gegensatz zu den anderen Kombinationen.

256 MB / 1MB = 256 Pages

Dies ist allerdings auch nichts Gutes, da somit die Gefahr für interne Fragmentierung besteht.

Die **erste Parameter Kombination** ist auch unrealistisch, da die Kombination viel zu klein ist. Denn 8 Bit pro Page ist zu klein.

18.4

Fehler 1: Virtueller Arbeitsspeicher größer als physischer Arbeitsspeicher

./paging-linear-translate.py -P 1m **-a 20m -p 10m** -v -s 3 -c

```
Error: physical memory size must be GREATER than address space size (for this simulation)
```

Dieser Fehler ist in dem Programm abgedeckt, denn der Physische Arbeitsspeicher muss immer größer sein, als der virtuelle Arbeitsspeicher.

Fehler 2: Page größer als virtueller Arbeitsspeicher

./paging-linear-translate.py -P **4m -a 2m** -p 512m -v -s 3 -c

```
Virtual Address Trace
Traceback (most recent call last):
  File "./paging-linear-translate.py", line 174, in <module>
    if pt[vpn] < 0:
IndexError: array index out of range
```

Dieser Fehler ist nicht abgedeckt, allerdings stürzt das Programm ab mit der Begründung: **Array index out of range**
Page darf nicht größer sein als der virtuelle Arbeitsspeicher

Fehler 3: Page, physischer oder virtueller Arbeitsspeicher mit der Größe 0 oder kleiner

./paging-linear-translate.py -P 4m **-a 0** -p 512m -v -s 3 -c

```
Error: must specify a non-zero address-space size.
```

Die Größer darf logischerweise nicht 0 sein.

Fehler 4: Der Adressspeicher ist kein Vielfaches von einer Page

./paging-linear-translate.py -P **2m -a 255** -p 512m -v -s 3 -c

```
Error in argument: address space must be a power of 2
```

Dies muss jedoch so sein, sonst ist Platz nicht verwendbar und das ist nicht erlaubt.