Hugepages, NUMA or nothing on Linux?

Daniel Hillinger
Value Transformation Services S.r.l. Zweigniederlassung Deutschland
München

Schlüsselworte

Memory; Arbeitsspeicher; NUMA; Hugepages

Einleitung

Speicherarchitekturen wie NUMA (Non-Uniform Memory Access) oder Hugepages bieten Vorteile und Nachteile. Wo liegen aber die Unterschiede, die für eine verbesserte Performance entscheidend sind? Für diese Frage gilt es erstmal einen Blick auf die Grundlagen zu werfen, um dann Unterschiede und Voraussetzungen zu erkennen. Gut zu wissen, welche Features dabei von Oracle unterstützt werden und was sie können. Für Datenbankadministratoren bietet dies eine Entscheidungshilfe, wenn es um die eigenen Datenbanken geht.

Hugepages

Normalerweise wird Memory in 4KB großen Blöcken verwaltet, sogenannte Pages. Wie der Name Hugepages schon sagt, sind dies größere Pages. Bei den meisten Linux-Distributionen sind Hugepages 2MB groß. Dadurch müssen, um den Faktor 512, weniger Pages verwaltet werden.
Das bedeutet in der Praxis: bei einer SGA von 10 GB braucht man 2621440 Pages, bei Hugepages nur 5120.

Nicht nur die Datenbank profitiert von der geringeren Anzahl an Pages, auch das restliche Betriebssystem hat einen Gewinn davon, da die Pagetable schneller durchsucht werden kann. Ein weiterer Vorteil ist, dass Hugepages nicht ausgelagert werden können. Damit können sie auch bei einer großen Memory-Benutzung nicht vom Arbeitsspeicher auf die Festplatte ausgelagert werden. Der kswapd ist somit deutlich weniger beschäftigt, da er die Hugepages nicht zu prüfen braucht.

Zu beachten ist dabei,dass bis Datenbank Version 11.2.0.2 keine Hugepages verwendet werden, wenn nicht die komplette SGA hineinpasst. Passt die SGA nicht hinein, kann es gefährlich werden, da der gesamte Arbeitsspeicher dann eventuell nicht mehr ausreicht.
Der Parameter „use_large_pages=[true|false|only]“ steuert das Verhalten der Datenbank.
Folgende Meldung ist im alert.log zu sehen, wenn Hugepages genutzt werden :

```
****************** Huge Pages Information *****************
Huge Pages memory pool detected (total: 33280 free: 32222)
DFLT Huge Pages allocation successful (allocated: 20481)
***********************************************************
```

Ab Version 11.2.0.3 werden so viele Hugepages wie möglich verwendet. Auch hier findet man im alert-log die notwenigen Informationen.

```
Total Shared Global Region in Large Pages = 4096 MB (85%)
```

NUMA

NUMA (Non-Uniform Memory Access) hingegen ist ein ganz anderer Ansatz. Dieser muss von der Hardware unterstützt werden und ist nur bei Multi-CPU-Systemen möglich. Viele ältere Systeme unterstützen kein NUMA. Diese ältere Architektur wird Symmetric-Multi-Processor (SMP) oder UMA (Uniform Memory Access) genannt. Dabei sind alle CPUs über den internen Bus verbunden, an diesem Bus ist auch der Memory Controller angeschlossen. Daraus ergibt sich für jede CPU die gleiche Zugriffszeit auf den Arbeitsspeicher.

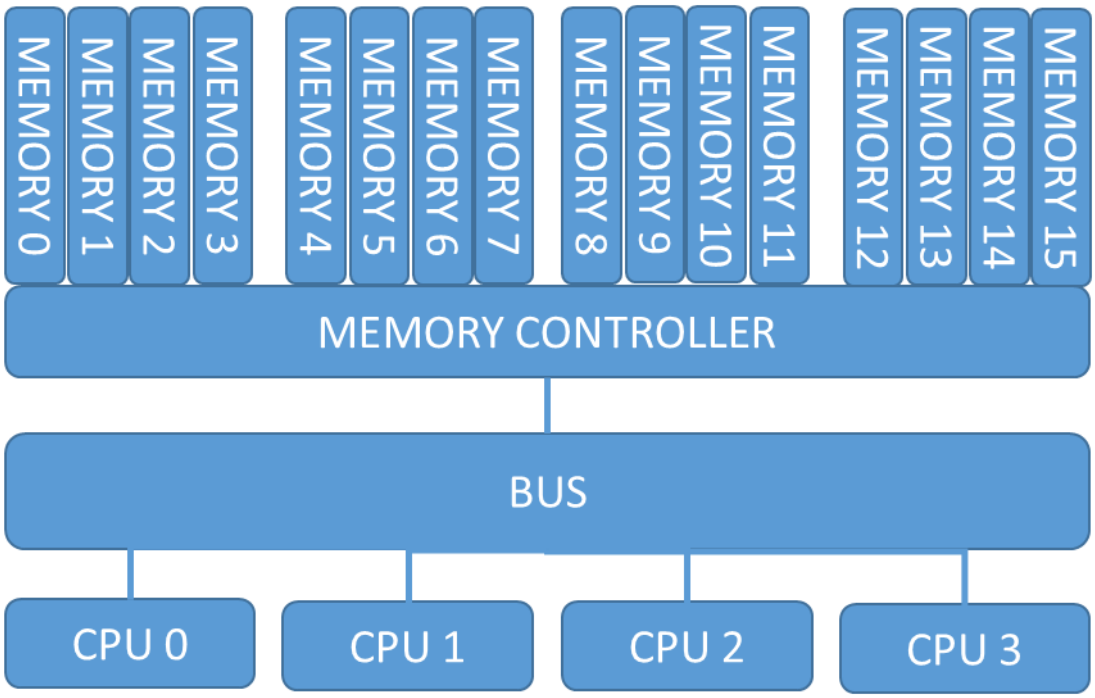

ABB.1: Symmetric Multi-Processor (SMP) Architektur

Bei NUMA hingegen ist der Memory Controller in die CPU integriert und deswegen pro CPU in lokal und remot Memory geteilt. Der lokale Memory hat sehr geringe Antwortzeiten. Hingegen ist der entfernte Arbeitsspeicher gleich schnell zu erreichen, wie bei der UMA-Architektur. In der folgenden Darstellung ist eine NUMA-Architektur mit 4 CPUs abgebildet. Der lokale Speicher für CPU 1 ist in diesem Fall Memory 4-7.

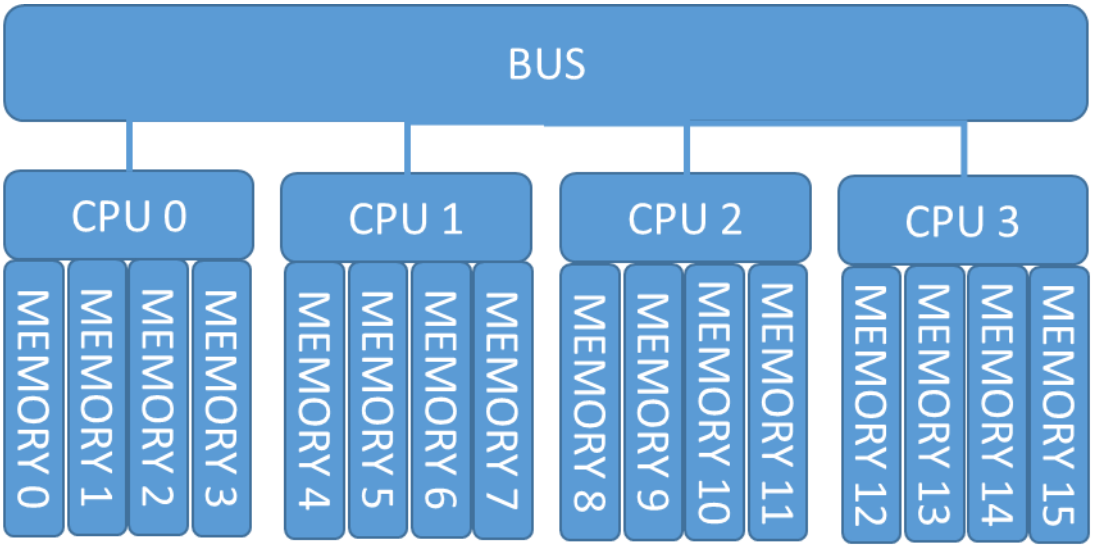

ABB.2: NUMA Architektur

Natürlich reicht es nicht aus, wenn nur die Hardware NUMA unterstützt. Auch das Betriebssystem und die Software muss NUMA klar kommen. Linux unterstützt NUMA seit der Kernelversion 2.6-14. Es wurde aber auch zurück portiert für Version 2.4, allerdings mit einer leicht veränderten Funktionsweise. Oracle unterstützt NUMA seit 10.2.0.4 und 11.1.0.7. Dabei ist NUMA standardmäßig aktiviert und lässt sich nur mit dem Patch 8199533 ausschalten. Aktivieren lässt es sich anschließend wieder mit dem Parameter „_enable_NUMA_optimization=TRUE“.

Informationen über Ihr System

Mit dem folgenden Kommando, lässt sich feststellen ob Ihr System NUMA unterstützt.

```
$ numactl --hardware
available: 4 nodes (0-3)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 40 41 42 43 44 45 46 47 48 49
node 0 size: 262122 MB
node 0 free: 175478 MB
node 1 cpus: 10 11 12 13 14 15 16 17 18 19 50 51 52 53 54 55 56 57
58 59
node 1 size: 262144 MB
node 1 free: 176411 MB
node 2 cpus: 20 21 22 23 24 25 26 27 28 29 60 61 62 63 64 65 66 67
68 69
node 2 size: 262144 MB
node 2 free: 175004 MB
node 3 cpus: 30 31 32 33 34 35 36 37 38 39 70 71 72 73 74 75 76 77
78 79
node 3 size: 262144 MB
node 3 free: 174307 MB
node distances:
node   0   1   2   3
  0:  10  11  11  11
```

```
1:  11  10  11  11
2:  11  11  10  11
3:  11  11  11  10
```

Der Output kann sich stark unterscheiden, je nach Linux-Distribution und Version. Wenn hierbei nur ein Node angezeigt wird, ist die Hardware nicht NUMA fähig.

Bei Hugepages gibt es einiges mehr zu beachten. Konfiguriert werden sie als Kernelparameter in der /etc/sysctl.conf. Der Parameter dafür heißt „vm.nr_hugepages“ und lässt sich auch im laufenden Betrieb ändern. Dies erfolgt mit dem gleichnamigen Befehl sysctl. Ob man sie erweitern kann, hängt natürlich von der Fragmentierung des Arbeitsspeichers ab.
Wenn Hugepages online erweitert werden, sollte das nicht dem Zufall überlassen sein. Daher ist die Prüfung in /proc/buddyinfo sinnvoll. Hier lässt sich konkret feststellen, welche Erweiterung mit Hugepages möglich ist.

```
$                          cat                              /proc/buddyinfo
Node 0, zone DMA0     1     0     1     1     0     1     …
Node 0, zone DMA32    9     6     10    9     9     8     …
Node 0, zone Normal   7941  5363  905   429   269   195   …
Node 1, zone Normal   3978  2228  654   366   188   130   …
Node 2, zone Normal   9594  4502  1397  695   359   149   …
Node 3, zone Normal   3297  3081  1732  287   132   111   …
```

Als Beispiel sind im Node 1 noch 3978 Pages der Größe 4K frei und 2228 * 2 ^ (1* 4K). Daraus ergibt sich das Muster X* 2^(Spaltennummer * Speichergröße (4K)). Für Hugepages sind nur zusammenhängende Blöcke von 2MB oder größer interessant. Diese Informationen befinden sich in den Spalten 10 und 11. Online erweiterbare Hugepages: (Spalte 10 + Spalte 11 * 2) * 2MB

Kompatibilität

Die PGA kann nicht mit Hugepages umgehen. Das ist der Punkt, den man immer im Hinterkopf behalten muss, da sich daraus alles Weitere ableiten lässt. Ein einfaches Beispiel: AMM (Automatic Memory Management) verwaltet SGA und PGA gemeinsam und verschiebt Memory zwischen den beiden Bereichen. Deswegen können hier Hugepages nicht eingesetzt werden.

	Hugepages	NUMA
AMM	nein	ja
ASMM	ja für SGA	ja
Manuelles Memory Managment	ja für Komponenten der SGA	ja

TAB.1: Komatibilitstmatrix

Theoretisch kann man auch beides, Hugepages und NUMA, gleichzeitig verwenden. Oracle hat das bisher aber nicht implementiert.

Fazit

Ich find es schade, dass Oracle nicht Hugepages und NUMA gleichzeitig implementiert hat, da dies den größten Performancegewinn gebracht hätte. Technisch wäre dies möglich.

Die NUMA-Implementierung ist ein großer Schritt in die richtige Richtung. Allerdings sagt Oracle selbst, dass dieses Feature mit großer Vorsicht zu betrachten ist. Es sollte laut Oracle sehr gut getestet werden. Nur so kann gewährleistet werden, dass es den gewünschten Performancegewinn bringt und Probleme vermieden werden.

Die Verwendung von Hugepages ist hingegen stabil. Für jede Datenbank mit großer SGA (SGA> 10G) empfiehlt sich die Verwendung von Hugepages, da es viele Vorteil für die Performance wie auch den DBA bietet. Seit Version 11.2.0.3, wo auch nur ein Teil der SGA Hugepages sein können, hat das Feature nur noch Vorteile.

Kontaktadresse:

Daniel Hillinger
Value Transformation Services S.r.l.
Zweigniederlassung Deutschland
Am Tucherpark 12
D-80538 München

Telefon: +49 (0) 89-378 28737
E-Mail Daniel.Hillinger@v-tservices.unicredit.de