記事
Tomohiro Iwamoto · 2020年6月8日 24m read

大規模データベースの仮想化 - VMwareのCPUキャパシティプランニング

VMware vSphereで実行する大規模な本番データベースのCPUキャパシティプランニングについて、お客様やベンダー、または社内のチームから説明するように頼まれることが良くあります。

要約すると、大規模な本番データベースのCPUのサイジングには、いくつかの単純なベストプラクティスがあります。

- 物理CPUコア当たり1つのvCPUを計画する。
- NUMAを考慮し、CPUとメモリをNUMAノードに対してローカルに維持できるようVMの理想的なサイズを決定する。
- 仮想マシンを適正化する。 vCPUは必要な場合にのみ追加する。

このことから、通常いくつかの一般的な疑問が生まれます。

- ハイパースレッディングにより、VMwareでは物理CPUの2倍の数でVMを作成できます。 これはキャパシティが2倍になるということか？ できるだけ多くのCPUを使ってVMを作成すべきではないのか？
- NUMAノードとは？ NUMAに配慮する必要があるのか？
- VMを適正化する必要があるが、どうすれば適正化されたことがわかるのか？

こういった疑問につては、下の例を使って答えることにします。 ただし、ベストプラクティスは決定事項ではありません。 ときには妥協することも必要です。 たとえば、大規模なデータベースVMはNUMAノードに収まらない可能性が高く、それはそれでも良いのです。 ベストプラクティスは、アプリケーションと環境に合わせて評価・検証する必要のあるガイドラインです。

この記事は、InterSystemsデータプラットフォームで実行するデータベースを例として書いていますが、概念とルールは、一般的にあらゆる大規模（モンスター）VMのキャパシティとパフォーマンスプランニングにも適用されます。

仮想化のベストプラクティスとパフォーマンスとキャパシティ計画に関するその他の記事:
「InterSystemsデータプラットフォームとパフォーマンス」シリーズの「その他の記事」セクションを参照してください。

モンスターVM

この記事は主に、「ワイドVM」と呼ばれることもあるモンスターVMのデプロイについて書いています。 高トランザクションデータベースのCPUリソース要件は、こういったデータベースがモンスターVMにデプロイされることが多い事を意味します。

> モンスターVMとは、単一の物理NUMAノードよりも多い仮想CPUまたはメモリを備えたVMです。

CPUアーキテクチャとNUMA

現行のIntelプロセッサアーキテクチャには、NUMA（Non-Uniform Memory Architecture）アーキテクチャがありま

す。 たとえば、この記事の検証用に使用しているサーバーは次の構成になっています。

- CPUソケット2個。各ソケットに12コアのプロセッサを装着（Intel E5-2680 v3）。
- 256 GBメモリ（16 x 16 GB RDIMM）

各12コアプロセッサには、独自のローカルメモリ（128 GBのRDIMMとローカルキャッシュ）があり、同じホスト内のほかのプロセッサのメモリにもアクセスできます。 CPU、CPUキャッシュ、および128 GB RDIMMメモリを1つのパッケージとした12コアをそれぞれNUMAノードと呼びます。 別のプロセッサのメモリにアクセスするために、NUMAノードは高速インターコネクトで接続されます。

ローカルRDIMMとキャッシュメモリにアクセスするプロセッサで実行するプロセスは、別のプロセッサのリモートメモリにアクセスするためにインターコネクトを経由する場合に比べ、レイテンシが低くなります。 インターコネクト経由のアクセスではレイテンシが高くなるため、パフォーマンスが一定しません。 同じ設計は3つ以上のソケットを持つサーバーにも適用されます。 ソケットが4つあるIntelサーバーには、4つのNUMAノードがあります。

ESXiは物理NUMAを認識し、ESXi CPUスケジューラはNUMAシステムのパフォーマンスを最適化するように設計されています。 ESXiがパフォーマンスを最大化する方法の1つは、物理NUMAノードにデータの局所性を作成することです。 この記事の例では、vCPUが12でメモリが128 GB未満のVMがある場合、ESXiはそのVMを物理NUMAノードの1つで実行するように割り当てます。 このことから次のルールについて言うことができます。

> 可能であれば、CPUとメモリをNUMAノードに対してローカルに維持できるようにVMのサイズを決定する。

NUMAノードより大きなモンスターVMが必要な場合は、それでもかまいません。ESXiは非常に優れた機能によって、要件を自動的に計算して管理することができます。 たとえば、ESXiは、最適なパフォーマンスを得るために、物理NUMAノードにインテリジェントにスケジュールする仮想NUMAノード（vNUMA）を作成します。 vNUMA構造は、オペレーティングシステムに公開されます。
たとえば、2つの12コアプロセッサを搭載したホストサーバーと16
vCPUを備えたVMがある場合、ESXiは8つの物理コアを2つのプロセッサに使用して、VM vCPUをスケジュールするため、オペレーティングシステム（LinuxまたはWindows）に2つのNUMAノードが表示されます。

また、VMを適正化して、必要以上のリソースを割り当てないようにすることも重要です。リソースが無駄になり、パフォーマンスが低下する可能性があります。 VMを適正化すればNUMAのサイズ設定にも役立つだけでなく、特に低から中程度のCPU使用率のある24 vCPUのVMよりもCPU使用率の高い（ただし安全な）12 vCPUのVMの方が、より効率的でパフォーマンスの改善が得られます。特にこのホストに、スケジュールされる必要がありリソースを争っているほかのVMがある場合はなおさらです。 このこともルールをさらに強化する要因といえます。

> 仮想マシンを適正化する。

注意:
NUMAのIntel実装とAMD実装には違いがあります。 AMDには、プロセッサごとに複数のNUMAがあります。 顧客のサーバーでAMDプロセッサを見たのはしばらく前になりますが、それを使用している場合は、計画の一環としてNUMAのレイアウトを確認してください。

ワイドVMとライセンス

最適なNUMAスケジュール設定を得る、ワイドVMの構成は
2017年6月訂正: ソケットあたり1 vCPUでVMを構成します。
たとえば、24 vCPUのVMは、デフォルトでそれぞれが1つのコアを持つ24 CPUソケットとして構成されます。

> VMwareのベストプラクティスのルールに従ってください。

例については、VMwareブログに記載のこちらの記事を参照してください。

VMwareのブログ記事には詳細が説明されていますが、これを執筆したMark Achtemichukは次の経験則を推奨しています。

- 高度なvNUMA設定はたくさんあるが、デフォルトの設定を変更しなければいけないのは、まれなケースのみ。
- 仮想マシンvCPU数は、単一の物理NUMAノードの物理コア数を超えるまでは、必ず、ソケットあたりのコア数が反映されるように構成すること。
- NUMAノードにある物理コア数以上のvCPUを構成する必要がある場合は、最少数のNUMAノード間でvCPU数を均一に分割すること。
- 仮想マシンのサイズが単一の物理NUMAノードを超える場合、奇数のvCPU数を割り当てないこと。
- vCPUホットアド(Hot-add)は、vNUMAを無効にしてもかまわない場合にのみ有効にすること。
- ホストの物理コアの合計数以上に大きいVMを作成しないこと。

Caché ライセンスはコアを計数するため問題ではありませんが、Caché以外のソフトウェアやデータベースの場合は、VMに24ソケットを指定するとソフトウェアライセンスに違いが生じる場合があるため、ベンダーと確認する必要があります。

ハイパースレッディングとCPUスケジューラ

ハイパースレッディング（HT）の話が持ち上がるとよく聞くのが「ハイパースレッディングはCPUコア数を2倍にする」ということです。 これは明らかに物理的に可能なことではありません。コア数が物理的に増えたり減ったりすることはありえません。 ハイパースレッディングは有効にすべきものであり、有効になればシステムのパフォーマンスを向上させることができます。 期待値はおそらくアプリケーションパフォーマンスにおいて20%増かもしれませんが、実際の量はアプリケーションとワークロードによって決まります。
が、それでも2倍ではありません。

VMwareのベストプラクティスの記事
で説明したように、大規模な本番データベースVMをサイジングするには、vCPUに完全専用の物理コアがサーバーにあると仮定することから始まります。基本的に、キャパシティ計画を実施する際はハイパースレッディングを無視するということです。 たとえば:

> 24コアのホストサーバーの場合、利用可能なヘッドルームがあることを考えて、本番データベースに合計で最大24 vCPUを計画します。

ピーク処理期間のアプリケーション、オペレーティングシステム、およびVMwareのパフォーマンスをしばらく監視したら、より高い統合が可能であるかどうかを判断することができます。 ベストプラクティスの記事では、ルールを次のように述べました。

> 1つの物理CPU（ハイパースレッディングを含む）= 1つのvCPU（ハイパースレッディングを含む）

ハイパースレッディングでCPUが2倍にならない理由

Intel Xeonプロセッサ上のHTは、1つの物理コア上に2つの論理CPUを作成する方法です。 オペレーティングシステムは、2つの論理プロセッサに対して効率的にスケジュールを設定できます。論理プロセッサ上のプロセスまたはスレッドがIOなどを待機している場合、物理CPUリソースは、別の論理プロセッサによって使用されます。 ある時点において進行できるのは1つの論理プロセッサのみであるため、物理コアがより効果的に使用されていても、パフォーマンスは2倍にならないのです。

ホストBIOSでHTが有効化されている場合、VMを作成する際に、HT論理プロセッサあたりのvCPUを構成するこ

とができます。 たとえば、HTが有効になっている24物理コアのサーバーで、最大48 vCPUのVMを作成することができます。 ESXi CPUスケジューラは、最初に個別の物理コアで（NUMAを考慮しながら）VMプロセスを実行することにより、処理を最適化します。 モンスターデータベースVMに物理コア数以上のvCPUを割り当てるとスケーリングにメリットがあるのかについては、この記事の後の方で説明することにします。

co-stopとCPUスケジューリング

ホストとアプリケーションのパフォーマンスを監視した後、ホストCPUリソースのオーバーコミットが可能であると判断する場合があります。 これがよいことかどうかは、アプリケーションとワークロードに大きく依存します。 スケジューラと監視すべき重要なメトリックを理解することで、ホストリソースをオーバーコミットしていないことを確認できます。

たまに、VMが進行するには、VMのvCPUと同じ数の空き論理CPUが必要だということを聞くことがあります。 たとえば、12 vCPUのVMは、実行が進む前に12個の論理CPUが「利用可能」になるまで「待機」する必要があります。 ただし、バージョン3以降のESXiではそうでないことに注意してください。 ESXiは、アプリケーションのパフォーマンスを向上させるために効率的なスケジューリングを使用しています。

複数の協調動作するスレッドまたはプロセスは頻繁に同期するため、一緒にスケジューリングしない場合、操作のレイテンシが増加する可能性があります。 たとえば、スピンループ内で別のスレッドがスケジュールされるのを待機しているスレッドです。 最良のパフォーマンスを得るために、ESXiはできる限り多くの兄弟vCPUを一緒にスケジュールしようとします。 ただし、CPUスケジューラは、統合環境で複数のVMがCPUリソースを争う場合に、vCPUを柔軟にスケジュールすることができます。 兄弟vCPUが進行しないにも関わらず一部のvCPUが進行してしまうと大きな時間差（これをスキューと呼ぶ）が生まれてしまうため、この場合は先行のvCPUがそれ自体を停止するかどうかを決定します（co-stop）。 co-stop（またはco-start）するのは、VM全体ではなくvCPUであることに注意してください。 これは、リソースのオーバーコミットがある場合でもうまく機能しますが、CPUのオーバーコミットが多すぎると必然的にパフォーマンスに影響します。 後の例2で、オーバーコミットとco-stopの例を示します。

これは、VM間のCPUリソースの争奪戦ではありません。ESXiのCPUスケジューラの仕事は、CPUのシェア数、予約、限界といったポリシーを確実に守りながら、CPU使用率を最大化し、公平性、スループット、応答性、スケーラビリティを確保することにあります。 予約とシェアを使用して本番ワークロードに優先順を付ける話はこの記事の目的から外れてしまい、アプリケーションやワークロードの組み合わせによって異なります。 Caché固有の推奨事項がある場合は、これについて後で触れるかもしれません。 CPUスケジューラにはさまざまな要因がかかわっているため、このセクションではその表面をざっと紹介しています。 詳細については、この記事の最後に記載する参考情報にあるVMwareのホワイトペーパーやその他のリンクを参照してください。

例

さまざまなvCPU構成を説明するために、高トランザクションレートのブラウザベースの病院情報システムアプリケーションを使って、一連のベンチマークを実行しました。 VMwareが作成したDVDストアデータベースベンチマークの概念に似ています。

ベンチマークのスクリプトは、ライブの病院実装から得た観察とメトリックを基に作成されており、使用率の高いワークフロー、トランザクション、およびシステムリソースを最も使用するコンポーネントが含まれています。 ほかのホスト上のドライバーVMは、ランダム化された入力データを使用したスクリプトを設定されたワークフロートランザクションレートで実行することで、Webセッション（ユーザー）をシミュレートします。 レート 1x のベンチマークがベースラインです。 レートは段階的に増減できます。

データベースとオペレーティングシステムのメトリックに加え、ベンチマークデータベースVMのパフォーマンスを測定するのに適したメトリックは、サーバーで測定されるコンポーネント（またはトランザクション）の応答時間です。 コンポーネントとは、エンドユーザー画面の一部分などを指します。 コンポーネントの応答時間が増加すると、ユーザー側においてはアプリケーションの応答時間に悪影響がでます。 パフォーマンスの高いデータベースシステムは、一貫した高いパフォーマンスをエンドユーザーに提供できるものでなければなりません。 次のグラフでは、最も遅く使用頻度の高いコンポーネントの応答時間の平均を算出し、一貫したテストパフォーマンスとエンドユーザーエクスペリエンスの指標を測定しています。 平均的なコンポーネント応答時間は1秒未満であり、ユーザー画面は、1つのコンポーネント、または複雑な画面の場合は多くのコンポーネントで構成されています。

> 常にピーク時のワークロードと、アクティビティの予定外の急増に対応するためのバッファに対してサイジングを行っていることを覚えておきましょう。
> 私の場合は通常、平均80%のピークCPU使用率を目指しています。

ベンチマークのハードウェアとソフトウェアの全リストはこの記事の最後にあります。

例1. 適正化 - ホストあたり1つのモンスターVM

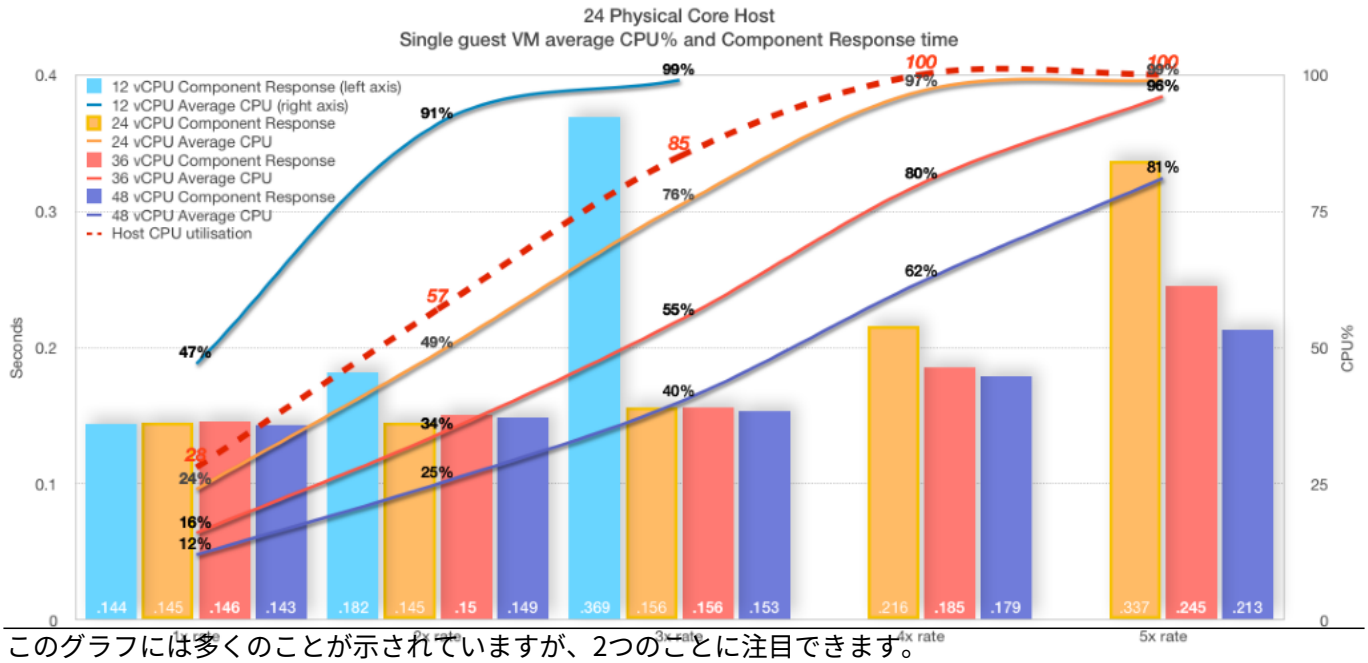
たとえば、24個の物理コアホスト上の24 vCPUのVMといったように、ホストサーバーのすべての物理コアを使用できるようにサイジングされたデータベースVMを作成することができます。 HAを得るためにCachéデータベースミラーで「ベアメタル」サーバーを実行したりオペレーティングシステムのフェイルオーバークラスタリングといった複雑なものを導入したりせずとも、データベースVMは、DRSやVMware HAなどの管理・HA向けvSphereクラスタに含まれます。

古い考え方で、5年後のハードウェア使用期間の最後に期待されるキャパシティに対してプライマリデータベースVMのサイズを決定するお客様を見てきましたが、上記で説明したように、適正化する方が有効と言えます。VMのサイズが過大に設定されていなければ、パフォーマンスと統合が改善され、HAの管理もより簡単になります。メンテナンスやホスト障害が発生した場合は、テトリスのように、データベースのモンスターVMを別のホストに移行して、そこで再起動しなければなりません。 トランザクションレートが大幅に増加すると予測される場合は、定期メンテナンス中に前もってvCPUを追加することができます。

> 「ホットアド」CPUオプションはvNUMAを無効にするため、モンスターVMには使用しないでください。

24コアホストでの一連の検証を示す次のグラフを考察してみましょう。 3x トランザクションレートは、この24コアシステムのスイートスポットであり、キャパシティプランニングの目標です。

- ホストでは単一のVMが実行している。
- 12、24、36、48 vCPUでのパフォーマンスを示すために4つのVMサイズを使用している。
- 各VMサイズに対し、トランザクションレート（1x、2x、3x、4x、5x）が実行された（可能な場合）。
- コンポーネント応答時間としてパフォーマンス/ユーザーエクスペリエンスを示している（棒グラフ表示）。
- 平均CPU%使用率はゲストVM（線グラフ表示）
- すべてのVMサイズのホストCPU使用率は、4x レートで100%に達した（赤い破線）。



このグラフには多くのことが示されていますが、2つのことに注目できます。

- 24 vCPUのVM（オレンジ）は、目標の3x トランザクションレートにスムーズにスケールアップしています。 3x レートにおいては、ゲスト内のVMは平均76%CPUに達成しています（ピーク時は約91%）。 ホストCPUの使用率はゲストVMとあまり変わりません。 コンポーネント応答時間は3xまでほぼ一定しているため、ユーザーは満足しています。 目標のトランザクションレートに関して言えば、このVMは適正化されていると言えます。

適正化についてはわかりましたが、vCPUの増加についてはどうでしょうか。これは、ハイパースレッディングを使用するということです。
パフォーマンスとスケーラビリティを2倍にすることは可能でしょうか。 短く言えば、答えは「ノー！」です。

この場合、この答えは、4x 以降のコンポーネント応答時間を見るとわかります。 割り当てられる論理コア数（vCPU）が多いほどパフォーマンスは「向上」しますが、一定しておらず、3x までのような一貫性がありません。 割り当てられるvCPU数が増えても、ユーザーは4x での応答時間が遅くなっていることを報告するでしょう。 vSphereが示したように、4xでは、ホストはすでに100%CPU使用率で横ばいになっていたことを思い出しましょう。 vCPU数が高い場合、ゲスト内CPUメトリック（vmstat）が100%使用率未満を示していたとしても、これは物理リソースには当てはまりません。 ゲストオペレーティングシステムは仮想化されていることを認識しないため、それに与えられたリソースに対して示しているだけであることに注意してください。 また、ゲストオペレーティングシステムはHTスレッドも認識しないため、すべてのvCPUは物理コアとして示されます。

ポイントは、データベースプロセス（3x トランザクションレートのCachéプロセスは200個以上ある）は非常にビジーであり、プロセッサを非常に効率的に使用しますが、ほかの作業をスケジュールするかこのホストにほかのVMを統合するための論理プロセッサにはあまり余裕がないということです。 たとえば、Caché処理の大部分はメモリ内で行われるため、IOの待機時間はあまりありません。 そのため、物理コアより多いvCPUを割り当てることはできますが、ホストの使用率がすでに100%に達しているため、あまり得がないのです。

Cachéは高いワークロードの処理に非常に優れています。 ホストやVMが100%のCPU使用率に達していても、アプリケーションは実行し続けており、トランザクションレートも増加し続けます。スケーリングは線形ではなく、応答時間が長引けば、ユーザーエクスペリエンスに影響があります。ただし、アプリケーションが「崖から落ちる」わけではなく、快適な場所でないにしても、ユーザーは作業を続行できるでしょう。 応答時間の影響をそれほど受けないアプリケーションであれば、限界まで押したとしても、Cachéは安全に動作し続けます。

> データベースVMやホストを100%CPUで実行するのは好ましいことではないことを覚えておいてください。 VMには予定外の急増や予測していなかった増加に対応できるキャパシティが必要であり、ESXiハイパーバイザーには、すべてのネットワーキング、ストレージなどでのアクティビティに使用するリソースが必要です。

私は常にピーク時のCPU使用率を80%で計画するようにしていますが、 それでも、vCPUのサイズを物理コア数までに設定するようにし、極端な状況でも論理スレッド上にESXiハイパーバイザー用のヘッドルームをいくらか残すようにしています。

> ハイパーコンバージド（HCI）ソリューションを実行している場合は、HCIのCPU要件もホストレベルで考慮する必要があります。 詳細は、HCIに関する前の記事
> を参照してください。 HCIにデプロイされたVMの基本的なCPUサイジングは、ほかのVMと変わりません。

独自の環境とアプリケーションですべての設定を確認・検証する必要があることを忘れないでください。

例2. オーバーコミットされたリソース

アプリケーションパフォーマンスは「遅い」にもかかわらず、ゲストオペレーティングシステムではCPUリソースに余裕があることが示される顧客サイトを見たことがあります。

ゲストオペレーティングシステムは仮想化を認識しないことに注意してください。 残念ながらvmstat（pButtonsの場合）が示すゲスト内のメトリックは当てにならないことがあるため、システムの状態とキャパシティを正確に理解するには、ホストレベルのメトリックとESXiのメトリック（esxtopなど）も取得する必要があります。

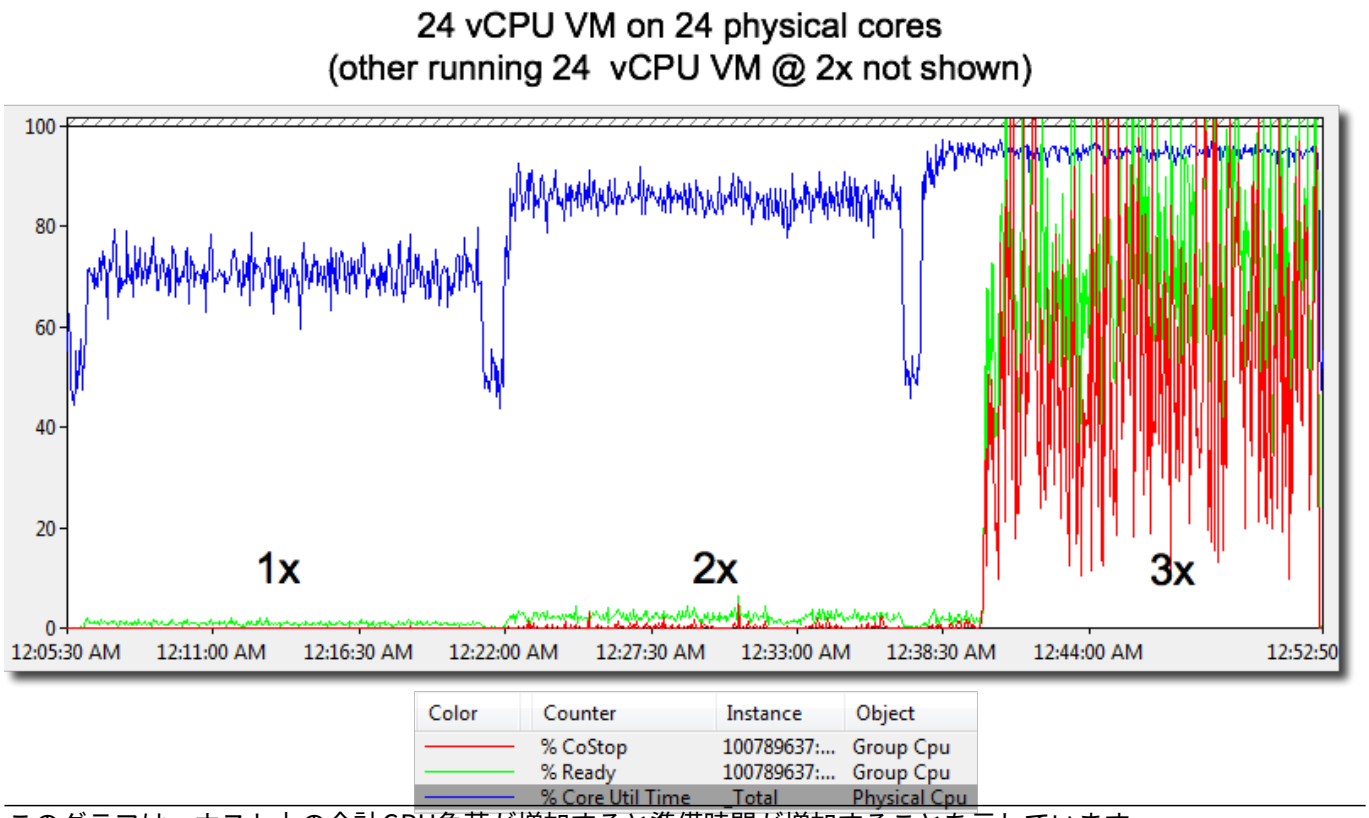
上のグラフからもわかるように、ホストが100%使用率を示す場合、ゲストVMの使用率は低く示されることがあります。 36 vCPUのVM（赤）は4x
レートで80%の平均CPU使用率を示していますが、ホストでは100%となっています。 公開後にほかのVMがホストに移行した場合や、きちんと構成されていないDRSルールによってリソースがオーバーコミットされている場合など、適正化されたVMでもリソース不足となることがあります。

主なメトリックを示すために、この一連の検証では以下のように構成しました。

- ホストで実行する2つのデータベースVM
  - 一定の2xトランザクションレートで実行する24vCPU（グラフには表示されていません）
  - 1x、2x、3xレートで実行する24vCPU（グラフに3つのメトリックが表示されています）

リソースを使う別のデータベースでは、3x レートにおけるゲストOS（RHEL 7）のvmstatは86%の平均CPU使用率を示しており、実行キューは平均25です。 ただし、このシステムのユーザーは、プロセスが遅くなるにつれコンポーネント応答時間が急増するため、大きな不満を訴えると考えられます。

次のグラフが示すように、co-
stopと
準備時間から
ユーザーパフォーマンスが
これほど悪い理由を読み取ることができます。 準備時
間（%Ready）とco-stop（%CoStop
）メトリックは、CPUリソースが目標の3xレートで大幅にオーバーコミットされていることを示しています。 ホストの実行レートが2x（other
VM）、かつこのデータベースVMは3xレートであるわけですから、特に驚くことでもないでしょう。



このグラフは、ホスト上の合計CPU負荷が増加すると準備時間が増加することを示しています。

> 準備時間とは、VMの実行準備は整っているが、CPUリソースが利用できないために実行できない時間です。

co-stopも増加します。 データベースVMを進行させられるのに十分な空き論理CPUがありません（上記のHTの項目で説明しました）。 その結果、物理CPUリソースの競合により処理が遅延しています。

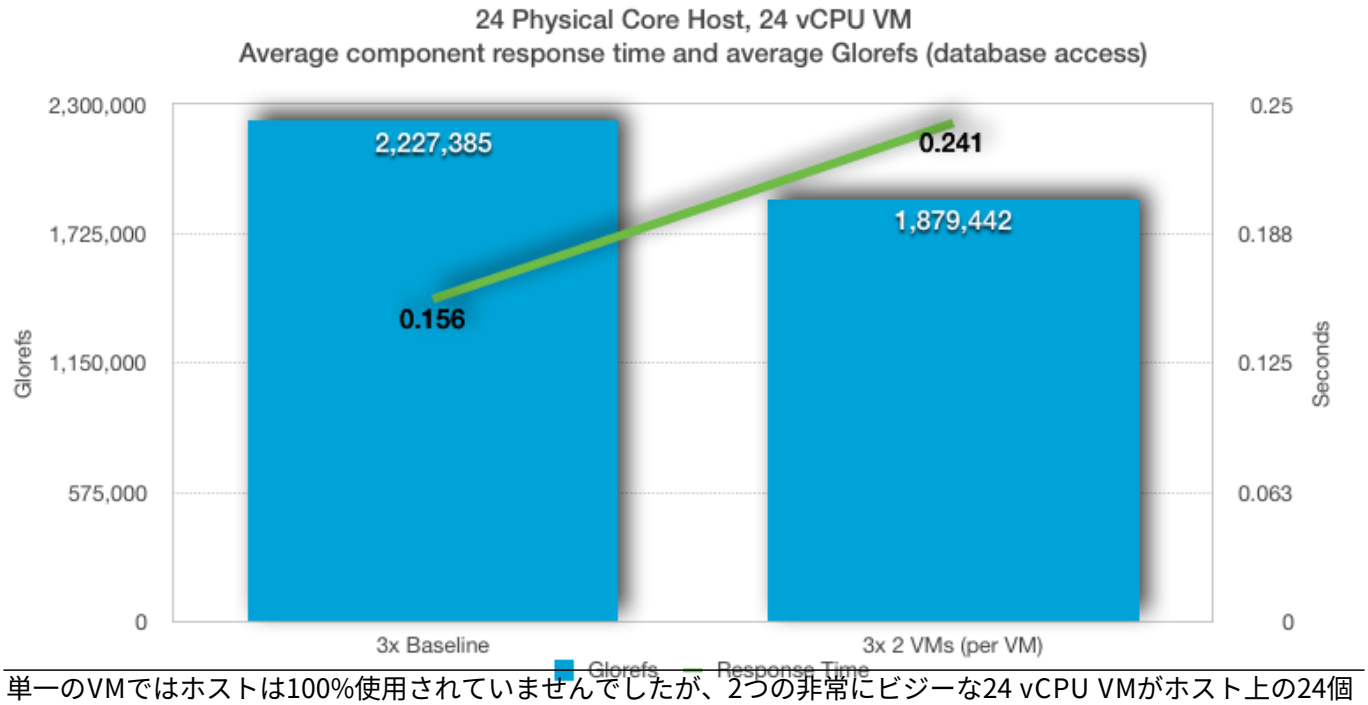
pButtonsとvmstatのサポートビューに、仮想オペレーティングシステムのみが表示されるという状況を顧客サイトで見たことがあります。 vmstatはCPUヘッドルームを示していましたが、ユーザーのパフォーマンスエクスペリエンスはひどいものでした。

ここでの教訓は、ESXiメトリックとホストレベルビューが利用できるようになるまで実際の問題が診断されなかったということです。一般的なクラスタCPUリソース不足によってCPUリソースのオーバーコミットが発生し、状況を悪化すべく、粗末なDRSルールによって高トランザクションデータベースVMがまとめてホストリソースに移行され、ホストリソースを圧迫していたのです。

例3. オーバーコミットされたリソース

この例では、3xトランザクションレートで実行する24 vCPUデータベースVMをベースラインとし、一定した3xトランザクションレートで2つの24 vCPUデータベースVMを使用しました。

平均ベースラインCPU使用率（上の例1を参照）は、VMが76%でホストが85%でした。 1つの24 vCPUデータベースVMが24個のの物理プロセッサをすべて使用しています。 2つの24 vCPU VMを実行すると、VMはリソースを争い、サーバーで48個の論理実行スレッドすべてを使用することになります。



単一のVMではホストは100%使用されていませんでしたが、2つの非常にビジーな24 vCPU VMがホスト上の24個の物理コアを使用しようとすると（HTを使用してでさえも）、スループットとパフォーマンスが大幅に低下するのがわかります。 Caché は利用可能なCPUリソースを非常に効率よく使用できますが、VMあたりのデータベーススループットは16%低下し、さらに重要なことに、コンポーネント（ユーザー）応答時間が50%以上増加しています。

最後に

この記事の目的は、一般的な疑問にお答えすることでした。 CPUホストリソースとVMware CPUスケジューラの詳細については、下の参考情報を参照してください。

システムの最後の一滴までパフォーマンスを絞り出す、特別に高度な調整やESXi設定がたくさんありますが、基本的なルールは非常にシンプルです。

大規模な本番データベースの場合:

- 物理CPUコアあたり1 vCPUで計画する。
- NUMAを考慮し、CPUとメモリをNUMAノードに対してローカルに維持できるようVMの理想的なサイズを決定する。
- 仮想マシンを適正化する。 vCPUは必要な場合にのみ追加する。

VMを統合する場合は、大規模なデータベースは非常にビジーであり、ピーク時にCPU（物理と論理）を大量に使用することに注意してください。 監視で安全だということがわかるまで、オーバーサブスクライブしてはいけません。

参考情報

- VMwareブログ「When to Overcommit vCPU:pCPU for Monster VMs」
- Introduction 2016 NUMA Deep Dive Series
- The CPU Scheduler in VMware vSphere 5.1

検証について

この記事で使用した例は、オールフラッシュアレイに接続された2つのDell R730プロセッサで構成されるvSphere

クラスタで実行しました。 例では、ネットワークまたはストレージにボトルネックはありませんでした。

- Caché 2016.2.1.803.0

PowerEdge R730

- 2x Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50 GHz
- 16x 16 GB RDIMM、2133 MT/秒、デュアルランク、x4 データ幅
- SAS 12 Gbps HBA外部コントローラ
- ハイパースレッディング（HT）有効

PowerVault MD3420、12G SAS、2U-24ドライブ

- 24x 24 960 GBソリッドステートドライブSAS読み取り処理集中型MLC 12 Gbps 2.5インチホットプラグドライブ、PX04SR
- 2 コントローラー、12G SAS、2U MD34xx、8Gキャッシュ

VMware ESXi 6.0.0ビルド2494585

- VMはベストプラクティスに合わせて構成されています。VMXNET3、PVSCSIなど

RHEL 7

- LargePages

ベースラインの1x レートの平均は700,000 glorefs/秒（データベースアクセス/秒）でした。 5x
レートの平均は、24 vCPUで3,000,000 glorefs/秒超でした。 検証では、一貫したパフォーマンスが達成されるまで慣らし運転を行い、その後で15分間、サンプルを取得し平均を算出しました。

> これらの例は理論を説明することだけを目的としているため、独自アプリケーションでは必ず検証する必要があります。

また、前の記事「
InterSystemsデータプラットフォームとパフォーマンス - VMバックアップとCachéのFreeze/Thawスクリプト
」もお読みください。

#システム管理 #デプロイ #Caché #InterSystems IRIS #InterSystems IRIS for Health #ドキュメント

ソースURL:
https://jp.community.intersystems.com/post/%E5%A4%A7%E8%A6%8F%E6%A8%A1%E3%83%87%E3%83%BC%E3%82%BF%E3%83%99%E3%83%BC%E3%82%B9%E3%81%AE%E4%BB%AE%E6%83%B3%E5%8C%96%C2%A0-%C2%A0vmware%E3%81%AEcpu%E3%82%AD%E3%83%A3%E3%83%91%E3%82%B7%E3%83%86%E3%82%A3%E3%83%97%E3%83%A9%E3%83%B3%E3%83%8B%E3%83%B3%E3%82%B0