ライブ仮想マシン移送の実現技術と最新動向について

山田 浩史
東京農工大学

第 19 回プログラミングおよびプログラミング言語ワークショップ（PPL 2017）

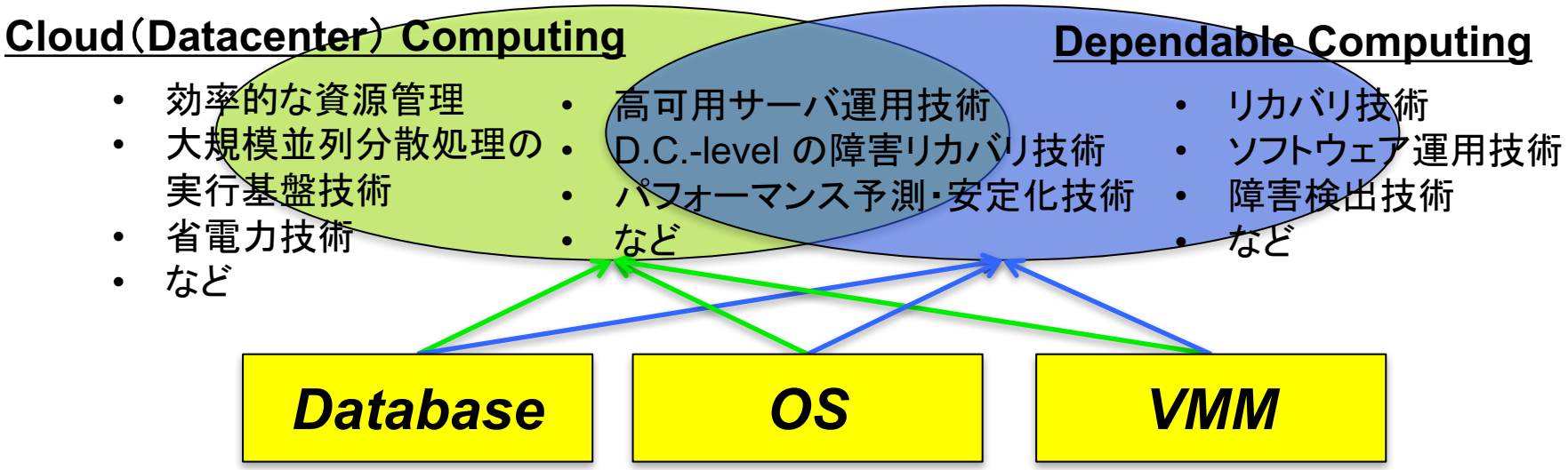
2017.03.10　　華やぎの章 慶山

Advanced Systems Group
（自己紹介）

- 次世代コンピュータサービスを実現する
  システムソフトウェア技術の探求
  - 想像・妄想ではなく、現実の課題を定量的・解析的にあぶり出す
  - 提案方式はソフトウェアとして実現し、動作させて検証する

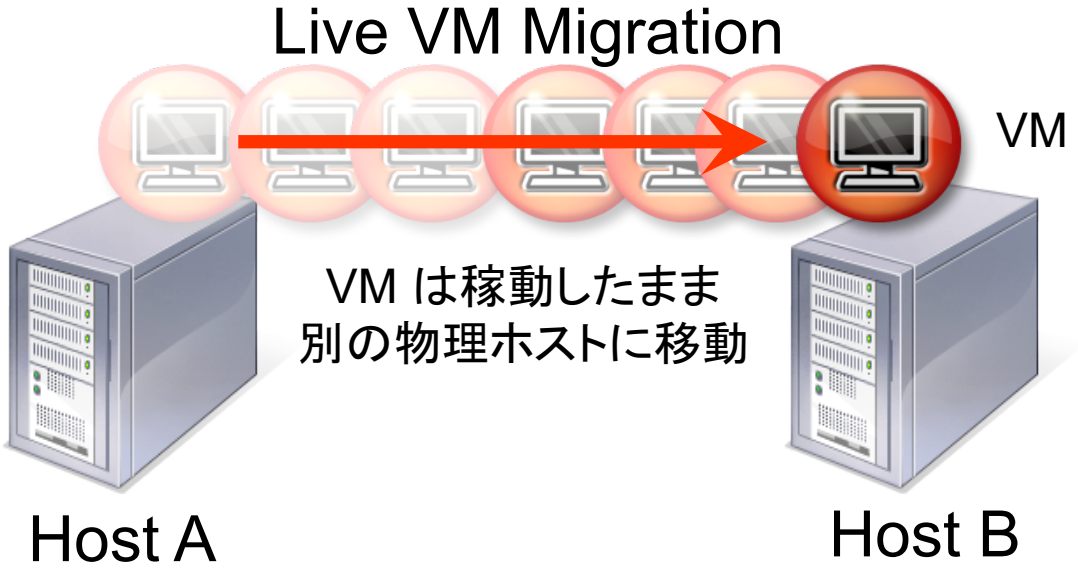
Real-World システムを対象とした、地に足の着いた研究を展開
～机上の理論だけではわからない“リアル”なトレードオフを解明～

APPROACH: 最新ハードウェアやアプリの動向を踏まえ、
理想的（***Ideal***）かつ実践的（***Pragmatic***）なシステムソフトウェア技術を生み出す



ライブ仮想マシン移送とは

- 仮想マシン（virtual machine，VM）を
  稼動させたまま、別の物理ホストへ移動させる技術



- 仮想化ソフトウェアでは一般的に提供されている
  - Xen、QEMU + KVM、VirtualBox、VMware ESXi、…

嬉しいこと

- VM の動的な再配置が簡単になる
  - ライブ VM 移送があると:
    - １つのコマンドで移送先を指定するだけ
  - ライブ VM 移送がないと:
    - VM のメモリイメージをキャプチャして、それを移送先へコピーして、移送先でイメージを復元して、しかもこれらを SLA に違反することなくやるためにタイミングを計って・・・
- 恩恵を受けるシナリオ
  - 物理マシンのメンテナンス
  - 負荷分散 [Wang+, VEE ’15], [Hermenier+, VEE ’09], [Wood+, NSDI ’07]など
  - 消費電力削減 [Govindan+, ASPLOS ’12], [Wang+, INFOCOM ’11], [Verma+, Middleware ’08]など

(僕が思う)ライブ VM 移送のいいところ

- 実運用で利用されている実践的な技術
  - Google のデータセンタではライブ VM 移送を用いて VM が管理されている
- ソフトウェア階層が深いがゆえに、様々なアプローチが考えられ得る
  - アプリ、OS、ハイパーバイザ、どこに手を入れるかは腕の見せどころ
- まだまだ発展途上な技術
  - 重いし、遅いし、改良の余地はまだまだある

(僕が思う)ライブ VM 移送のいいところ

- 実運用で利用されている実践的な技術
  - Google のデータセンタではライブ VM 移送を用いて VM が管理されている

ごちゃごちゃ言いましたが
単純におもしれぇ技術っす

  - ソフト、OS、ハイパーバイザ、どこに手を入れるかは腕の見せどころ

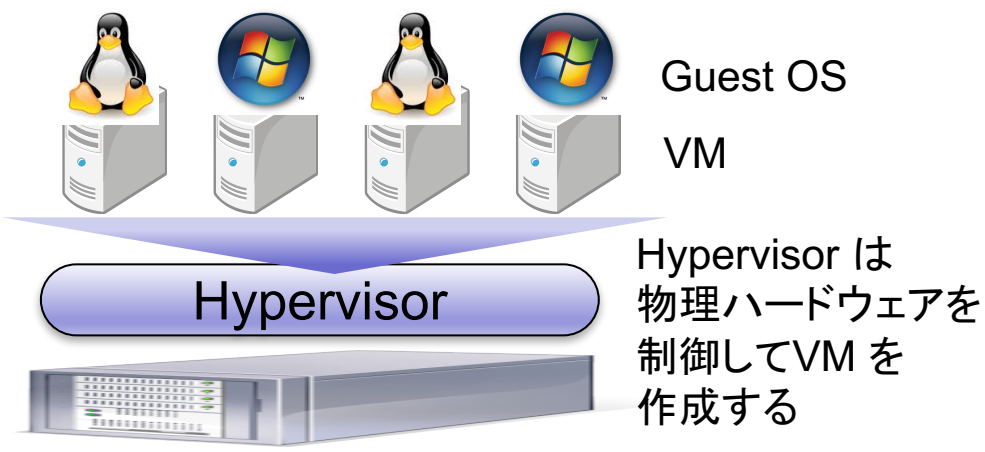
- まだまだ発展途上な技術
  - 重いし、遅いし、改良の余地はまだまだある

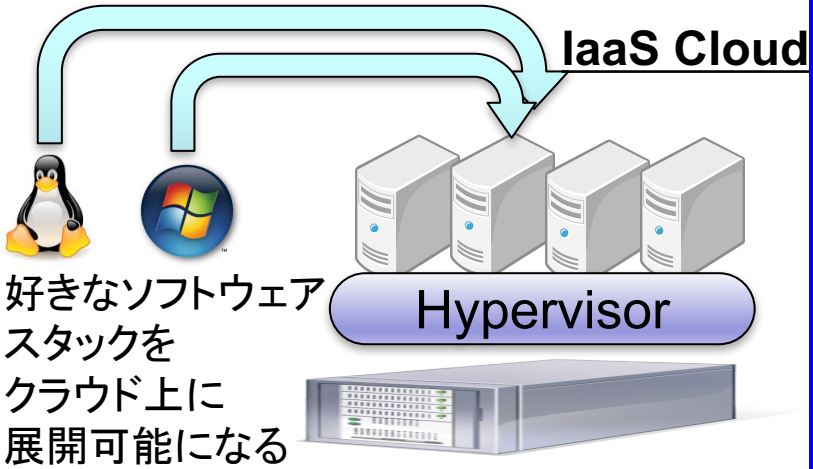
AGENDA

- 仮想化技術について
  - ハイパーバイザとその簡単な役割
- ライブ VM 移送の基礎
  - 3 種類のメモリ転送アルゴリズム
- ライブ VM 移送の実現手法の最新動向
  - 実現手法の研究成果
- ライブ VM 移送とアプリの関係
  - なんかよい技術ってないですか？

仮想化技術

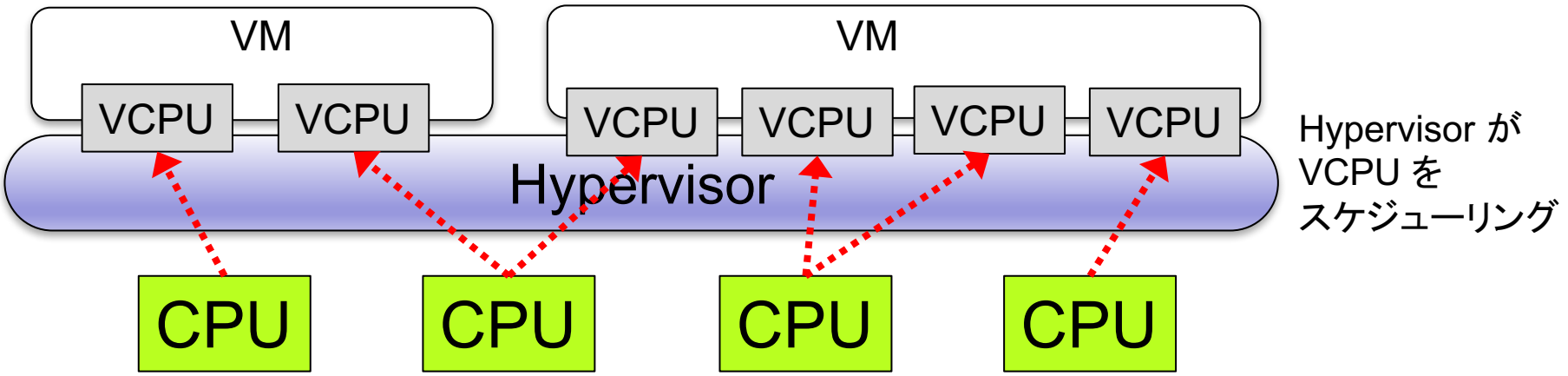
- 1 台の物理マシン上に仮想的なハードウェア（仮想マシン（VM））が利用できる
  - VM 上に OS を稼動させ、複数の OS を同時に実行できる
    - VM 上で稼動している OS を Guest OS と呼ぶ
- 仮想マシンモニタ（VMM、ハイパーバイザ（Hypervisor））と呼ばれるソフトウェアが VM を作り出す
  - 実ハードウェアを管理する
- クラウド環境を中心に利用されている
  - Amazon EC2: Xen をベースとした仮想化環境を提供
  - Google Compute Engine: KVM をベースとした仮想化環境を提供





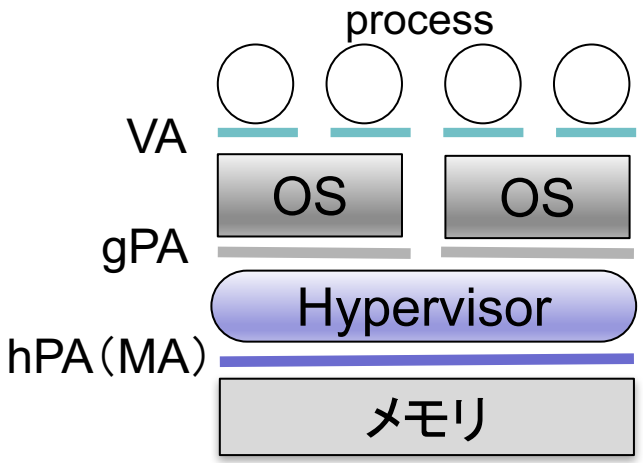
CPU の仮想化

- 各 VM の仮想 CPU（VCPU）をハイパーバイザが実 CPU にディスパッチ
  - プロセスと OS と同じような関係がVCPUとハイパーバイザ間で成立する
    - Amazon EC2 では 3 GHz の CPU を 3 つの small instance で時分割で共有している [Cu+, USENIX ATC '13]
  - ハイパーバイザは VCPU を管理する
    - 状態（Running, Idle, Block など）や保持している VM など
    - 例：Xen では struct vcpu という制御ブロックがある

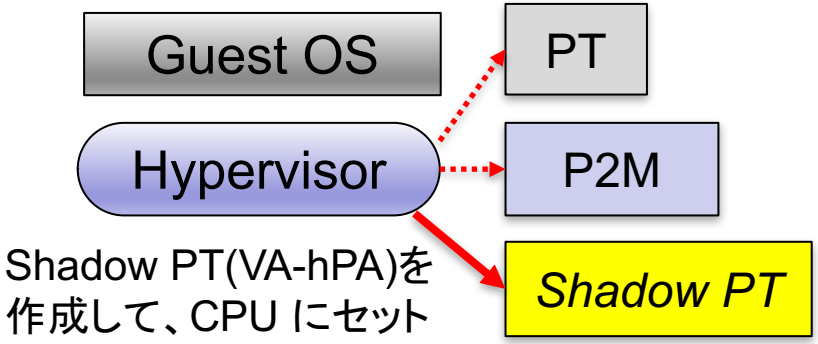


メモリの仮想化

- ハイパーバイザが実メモリへのマッピングを管理する
  - ゲスト OS: 仮想アドレス（VA）-ゲスト物理アドレス（gPA）の対応を保持
  - ハイパーバイザ: gPA-ホスト物理アドレス（hPA，MA）の対応を保持
    - P2M テーブルと呼ばれる
- ソフトウェアとハードウェアの仮想化支援を活用する方法がある
  - Shadow Paging
    - ゲスト OS のページテーブルのセットを補足して、VA–hPAのページテーブルを作成、セットする
  - Extended PT(Intel) / Nested PT(AMD)
    - ゲスト物理アドレスとホスト物理アドレスの対応表をセットする

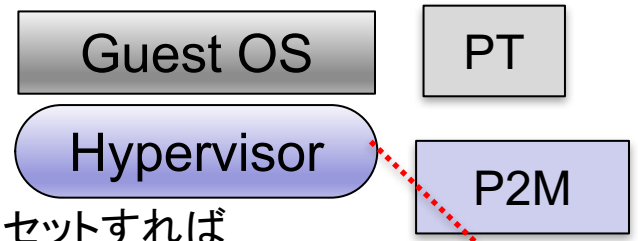


Shadow Paging



Shadow PT(VA-hPA)を作成して、CPU にセット

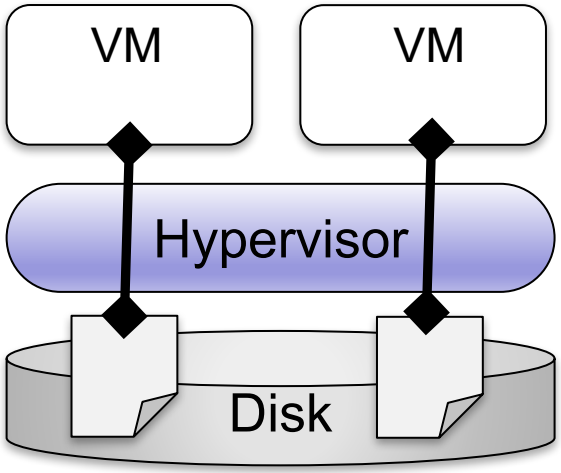
Virtualization support on CPU



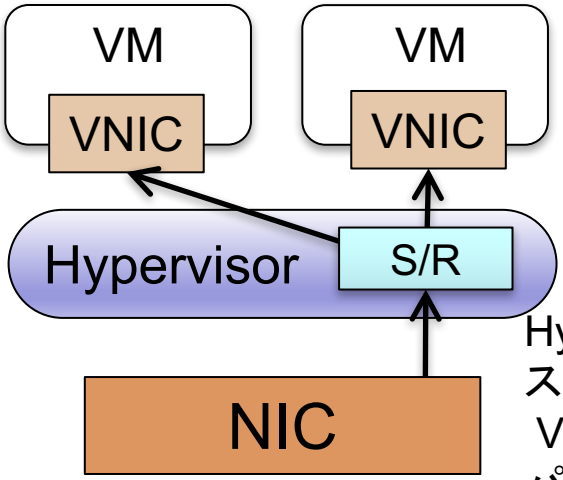
P2M をセットすれば VA-hPA の変換を CPU がやってくれる

デバイスの仮想化

- ハイパーバイザがデバイスの挙動をエミュレートする
  - ゲスト OS の I/O リクエストをトラップして、命令に応じて処理を実行する
  - 実デバイスを多重化して VM に提供する
- 多くのハイパーバイザがディスクと NIC をサポートしている
  - ディスク: ファイルに Guest OS のディスク書き込みを反映する
  - NIC: ソフトウェアスイッチ/ルータが VNIC にパケットを配送



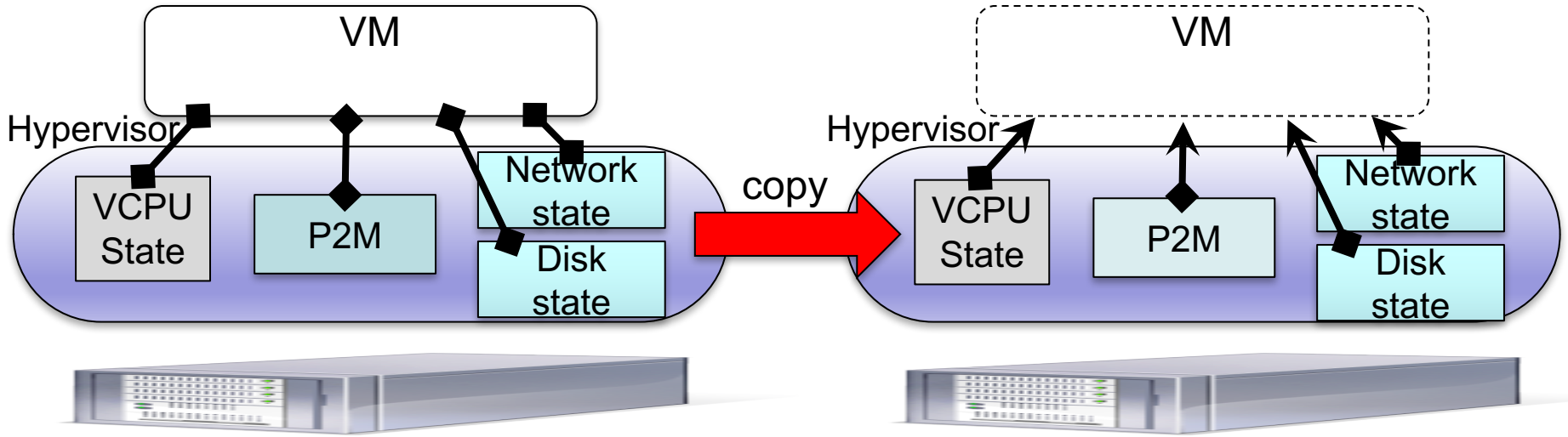
通常のファイルをディスクのように扱う



Hypervisor 内のスイッチ or ルータが VNIC にパケットを配送

ということは・・・

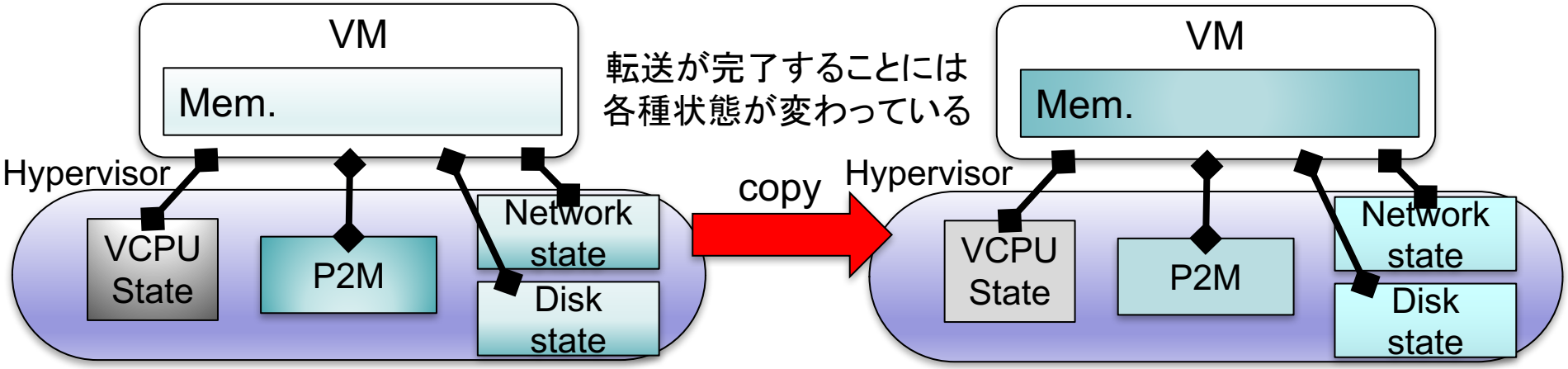
- VM の仮想ハードウェアの状態を他の物理ホストにコピーすれば、移送ができそう
  - ハイパーバイザが実行状況をすべて掌握してるため、移送先でそれらを復元すればよい
    - VCPU: VCPU の制御ブロックを移送先にコピーする
    - メモリ: メモリ内容をコピーしてとゲスト物理-ホスト物理のマップを再構築する
    - デバイス: 実行状態をコピーする



ライブに移送するのは難しい

- メモリ転送に時間がかかってしまう
  - 数GB ～数 100 GB のメモリを有する VM はざら
  - 移送先に転送している最中に状態が変わってしまう
    - VM は稼動しているため、VCPU やメモリの状態は転送中に変化

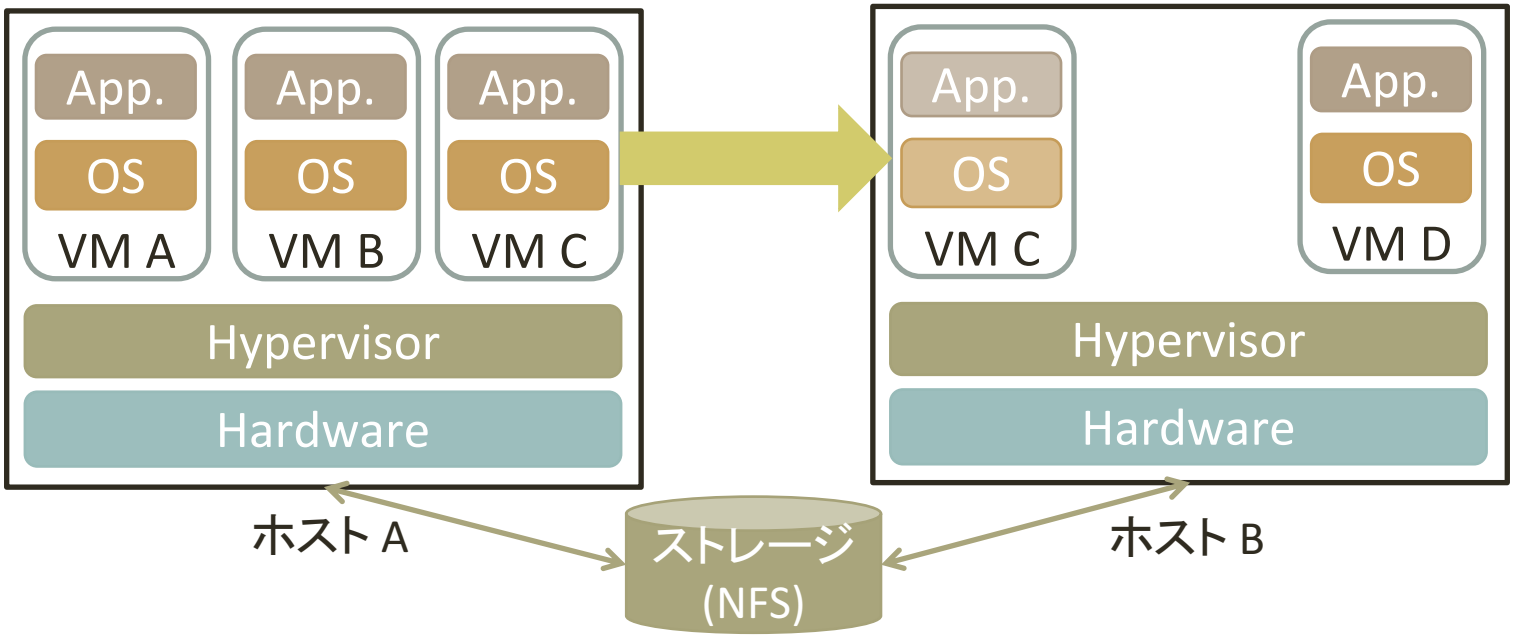
    ⇒ 同じ VM の状態を移送先で作ることができない



- Challenge: VM を稼動させたまま、同じ実行状態をいかに移送先で作成するか

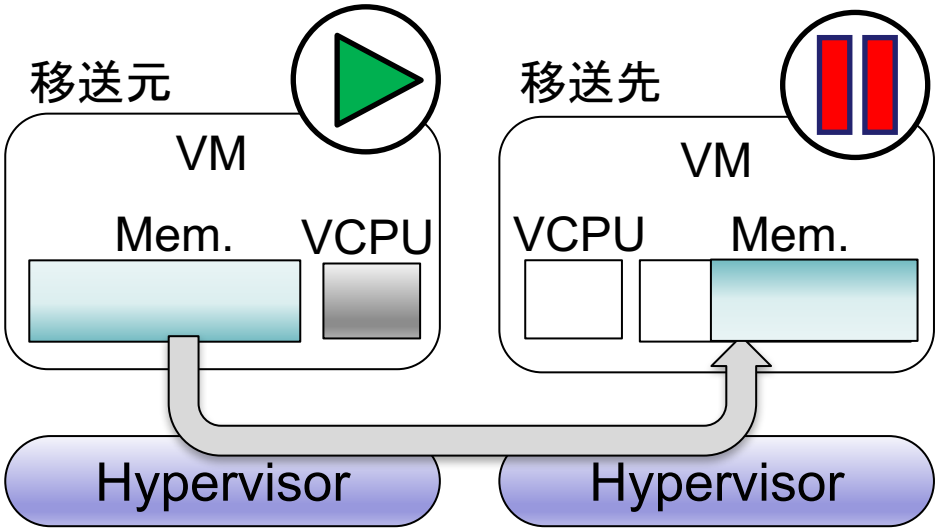
ライブ VM 移送の種類

- 3 種類の代表的な転送方法
  - Pre-copy、Post-copy、Hybrid（Pre-copy + Post-copy）
- 仮定: 仮想ディスクは共有ストレージにあるとする
  - 移送元・移送先から仮想ディスクにはアクセスできる
  - 仮想ディスクを含む移送は本発表の対象外とする

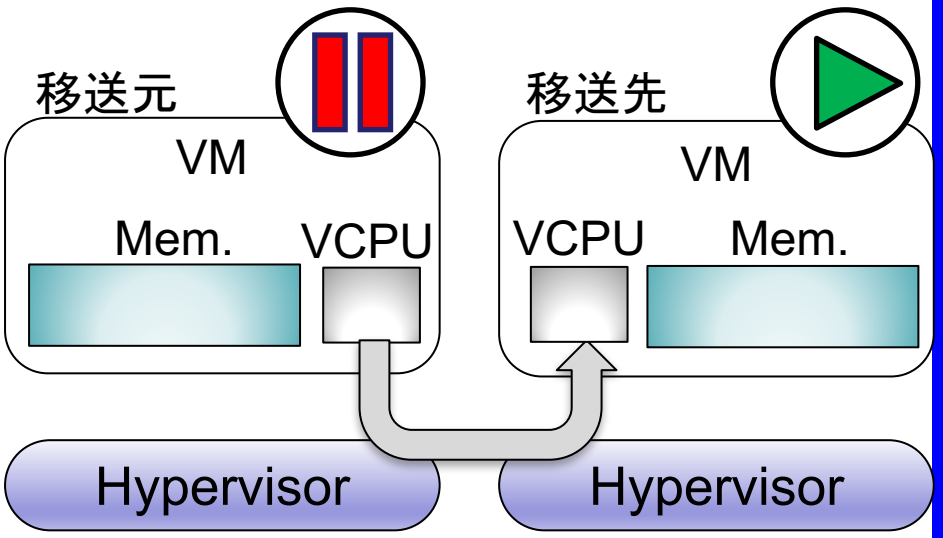


オーソドックスなやり方: Pre-copy

- メモリを転送したあとに移送先で VM を実行する [Clark+, NSDI ’04]
  1. メモリを移送先にコピーする
  2. メモリコピーが終わったら、他の状態（VCPU など）をコピーする
  3. 実行を移送先に移して、移送元の状態を破棄する



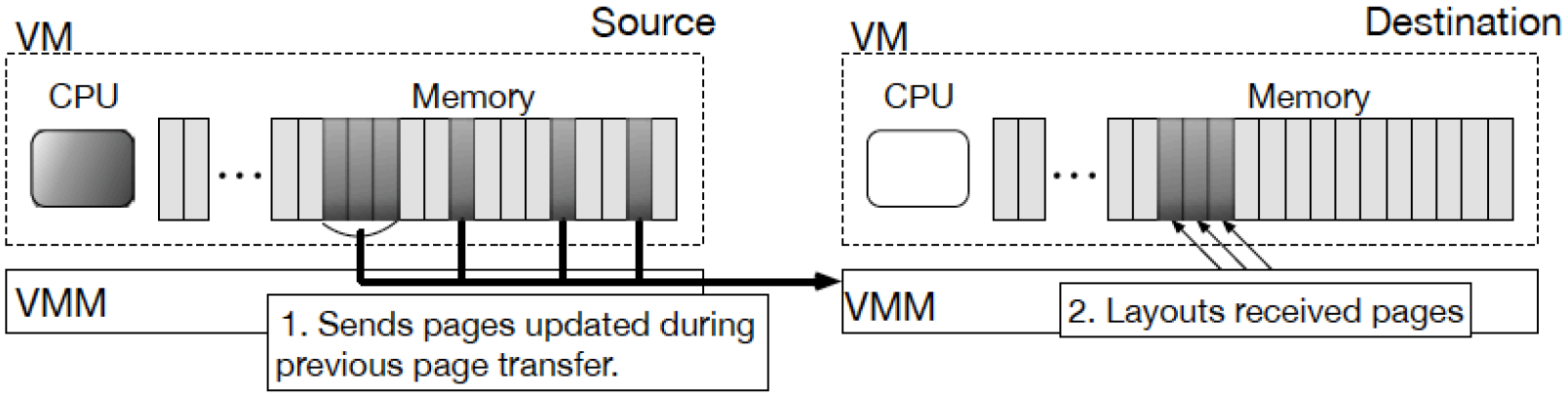
移送元で VM を動かしながら、メモリ内容を移送先にコピーする。



コピーが終了したら、VM を停止して、VCPU の状態をコピーする。その後、移送先で実行を再開。

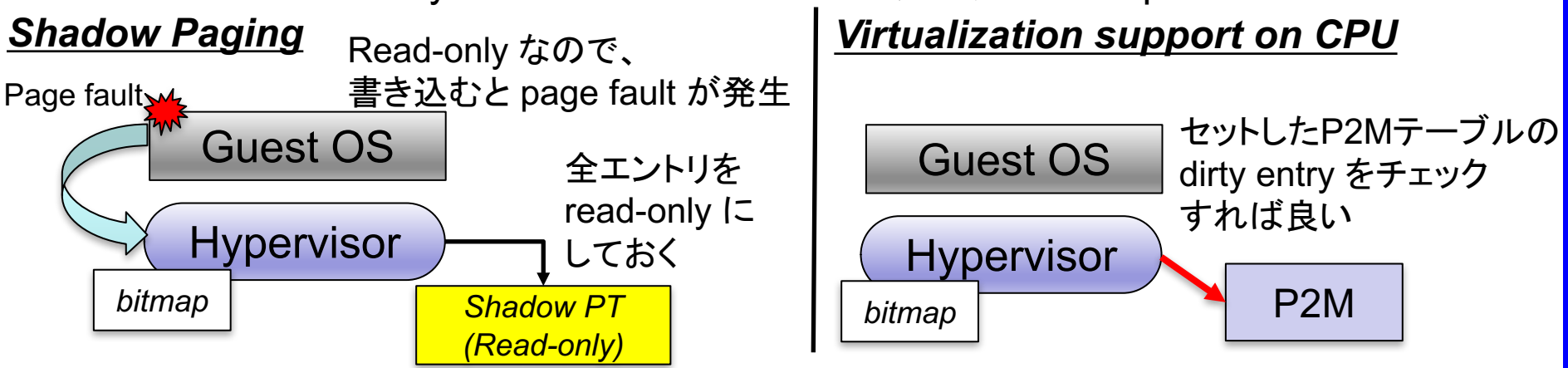
Pre-copy: 2 段階のメモリコピー

- Iteration Phase
  - 前回の転送時に dirty となったメモリページを転送する
    1. VM の全メモリを移送先に転送する
    2. 転送後に更新のあったページを検出する
    3. 検出したページを転送、2 へ
- Stop & Copy Phase
  - VCPU の状態を転送して、移送先に実行を移す
    1. VM を停止する
    2. VCPU の状態、更新のあったページを検出・転送する
    3. 転送後、VM を移送先で再開して移送元の資源を解放する



Dirty Page Tracking

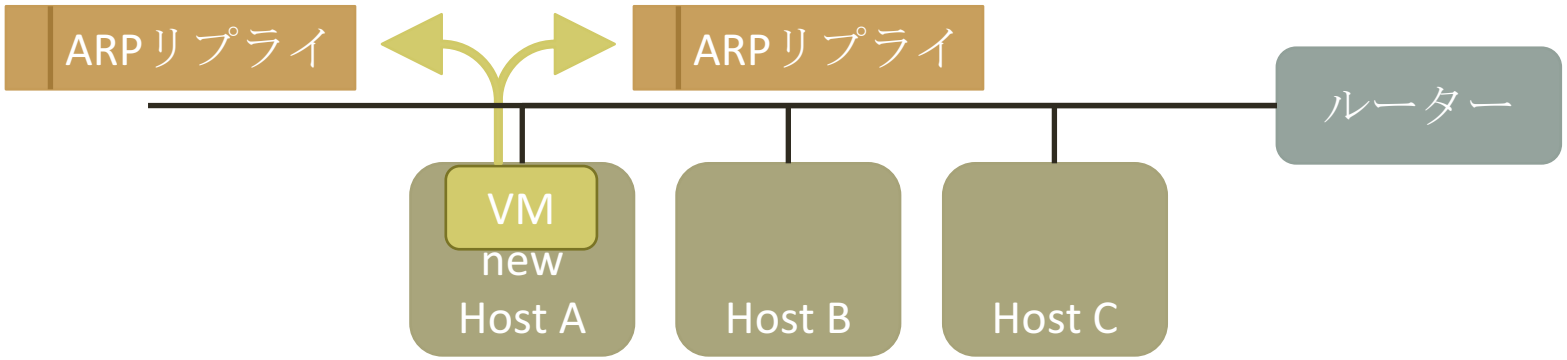
- 更新のあったメモリページを検出する
  - 多くの場合、bitmap で管理される（dirty bitmap）
    - ビットが立っているページを転送する
    - 各 Iteration の開始時に確認、クリアする
  - 更新の検出
    - Shadow Page Table を活用: ページを read-only にする
      - 書き込み違反を起こしたページに該当する bitmap を立てる
    - Intel EPT/AMD Nested PG を活用: dirty bit を確認する
      - Dirty bit が立っているページに該当する bitmap を立てる



IP アドレスの移送

- 移送先ホストから ARP リプライを
  ブロードキャストで送る
  - IP アドレスが新しい物理マシンに移動したことがわかる

  ⇒ クライアントとのコネクション維持を実現する



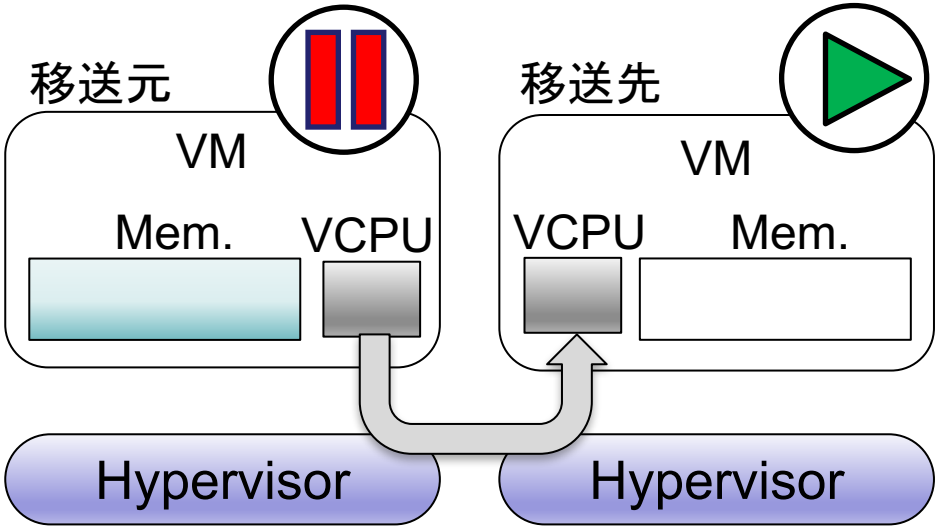
Pre-copy の問題点

- Write-intensive なアプリが稼動すると、移送時間が長期化しがち
  - 更新されるページが多いと、各 Iteration Phase での転送量が多くなる
  - Stop & Copy Phase での転送量も多くなるため、ダウンタイムも長くなる

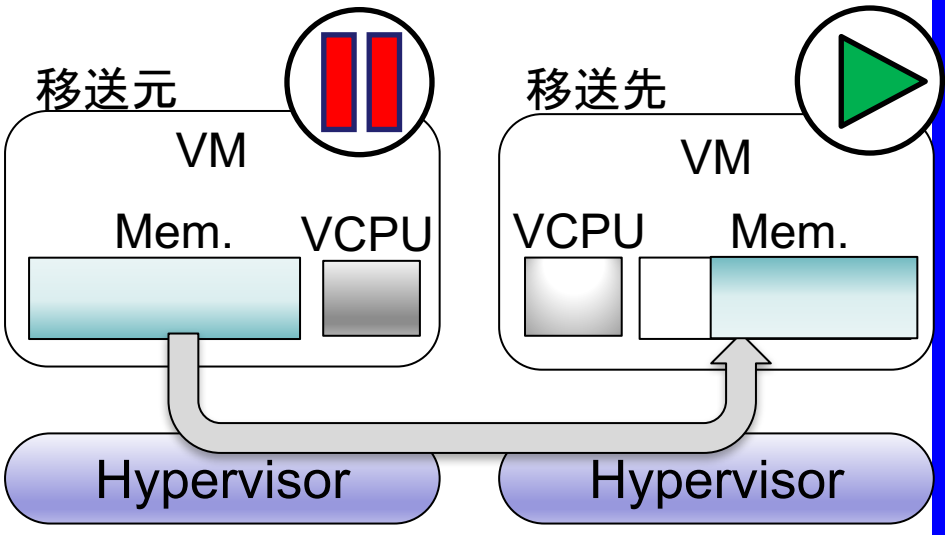
← 移送時間は短ければ短いほどよい
  - 再配置ポリシーで VM の配置を決めても状況が変わってしまう
  - ライブ VM 移送実行による CPU 使用率は意外と高い [Koto+, APSys '12]
    - 移送元、移送先の VM と CPU 競合が長期化する可能性

Post-copy

- 移送先で VM を開始して、順次メモリを転送する [Hines+, VEE ’09]
  1. メモリを移送先にコピーする
  2. メモリコピーが終わったら、他の状態（VCPU など）をコピーする
  3. 実行を移送先に移して、移送元の状態を破棄する



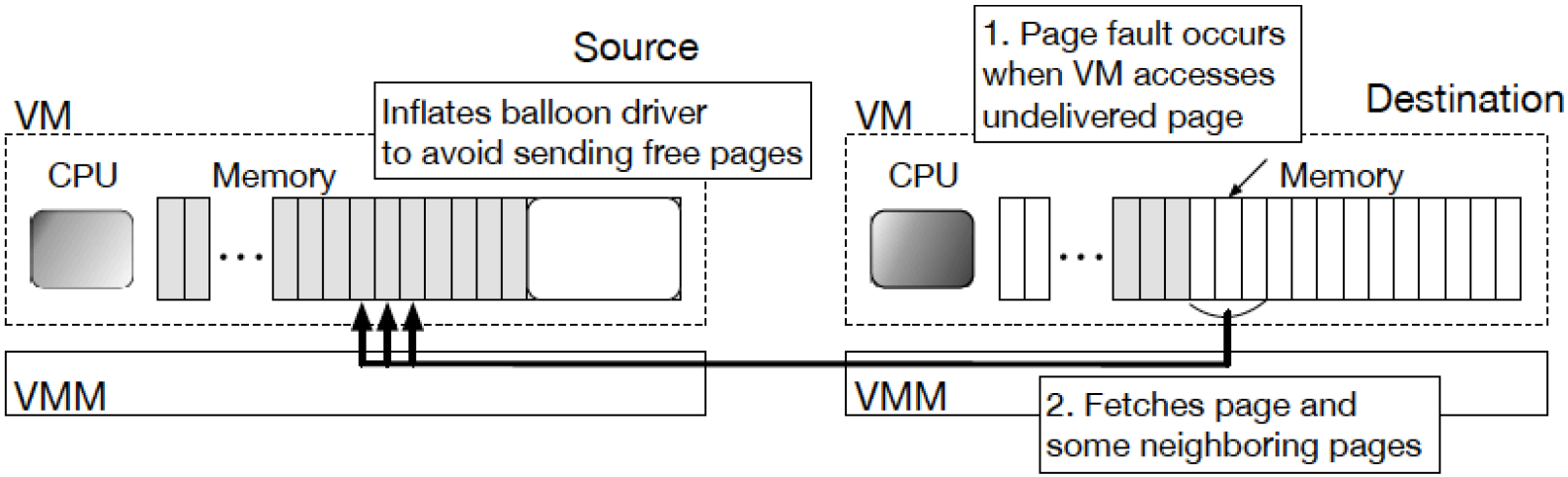
VM を停止して、VCPU の状態を移送先にコピーする。その後、移送先で実行を再開。



再開後、転送していないメモリを順次移送元から取得していく。

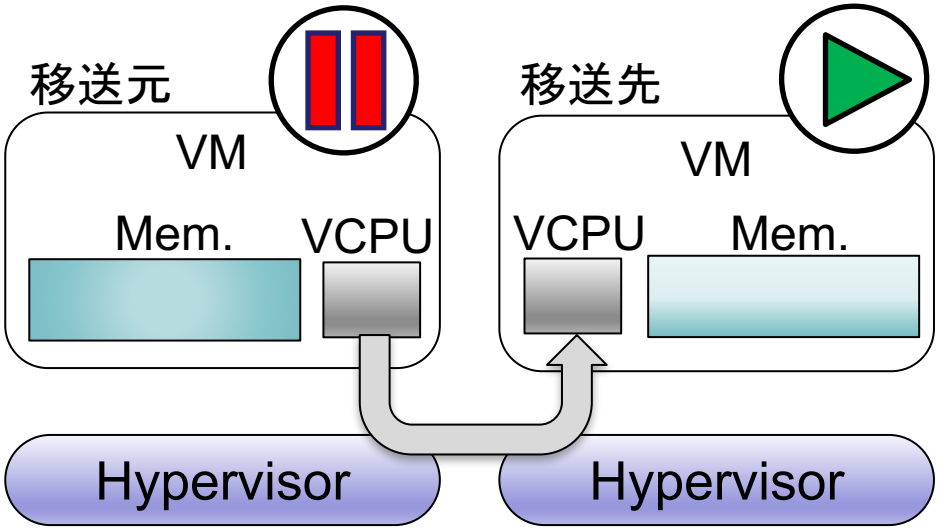
Post-copy: 2 段階のメモリコピー

- Stop & Copy Phase
  - VCPU の状態を転送して、移送先に実行を移す
    1. VM を停止する
    2. VCPU の状態、更新のあったページを検出・転送する
    3. 転送後、VM を移送先で再開する
- Memory Fetch Phase
  - 残りのメモリを移送元から移送先へ転送する
    - ページが必要になったら、そのページと周辺ページを転送する
      - 未転送のページへのアクセスはページフォールトが生じる
    - フリー領域は Balloon driver [Waldspurger, OSDI '02] を活用して転送しない

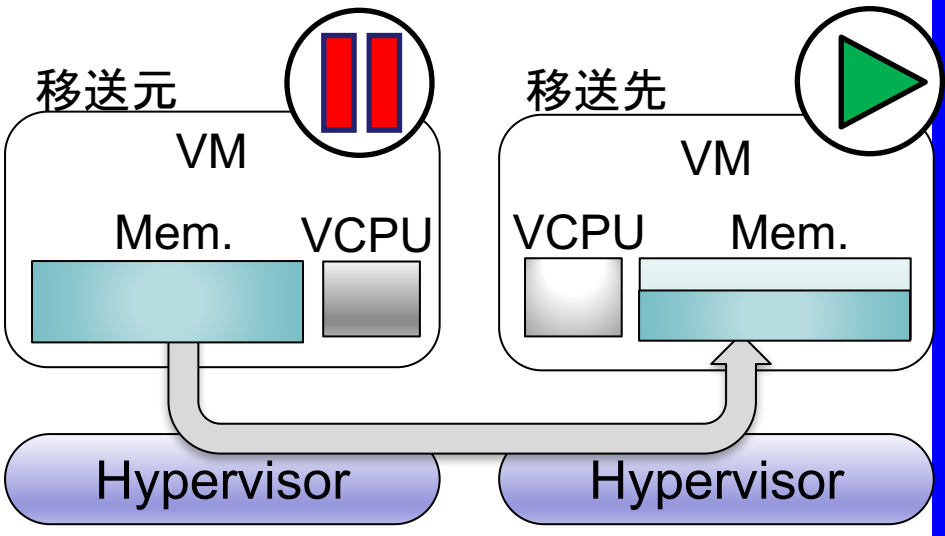


Hybrid: Pre-copy + Post-copy

- 両者のいいとこどりをしたようなメモリ転送手順
  1. VM の全メモリを転送する（Iteration Phase の 1 回目）
  2. VM を停止して、VCPU の状態を転送して
     移送先で VM を再開する（Stop & Copy Phase）
  3. 1.の転送中で更新のあったページを順次転送する
     （Memory Fetch Phase）



移送元から全メモリをまずは転送。その後、VM を停止し、VCPU の状態を転送して移送先で再開。



再開後、最初のメモリ転送で dirty となったページを順次転送していく。

各方式の比較

- 3 種類の手法は一長一短であり、適切に使い分けることが効果的
- Post-copy・Hybrid は移送の中断ができない
  - 実行状態が分離するため、中断すると VM が壊れる

  ⇒ 多くのハイパーバイザが Pre-copy をデフォルトとしている

Strategy	移送時間	メモリ転送量	VM 性能	信頼性	Description
Pre-copy	✗	✗	✔✔	✔	Write-intensive なワークロード下では移送が長期化しがち
Post-copy	✔✔	✔✔	✗	✗	移送直後はページの転送によるオーバヘッドが大きい
Hybrid	✔	✔	✔	✗	上記のいいとこどりだが、信頼性は post-copy の性質を引き継ぐ

ライブ VM 移送の実現技術

- 大きく 3 種類に分けられる
  - 多くが Pre-copy を題材としている

移送実行の高速化・効率化

[Nathan+, VEE '16]、
[Abe+, VEE '16]、
[Suetake+, APSys '16]、
[Hou+, EuroSys '15]、
[Mashtizadeh+, USENIX ATC '14]、
[Lu+, SYSTOR '13]、
[Chiang+, VEE '13]、
[Song+, VEE '13]、
[Cui+, VEE '13]、
[Koto+, APSys '12]、
など

移送可能な VM 設定を増やす

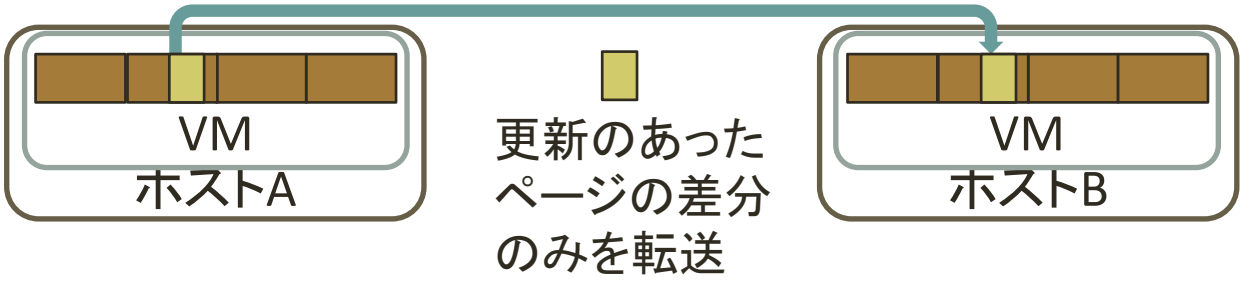
[Xu+, VEE '16]
[Ghorbani+, SoCC '14]、
[Gottschlang+, FHC '13]、
[Pan+, VEE '12]、
[Kadav+, WIOV '08]、
[Huang+, VEE '07]、
[Bradford+, VEE '07]
など

移送技術の応用

[Barker+, ICAC '14]、
[Zheng+, VEE '14]、
[Minhas+, VLDB '13]、
[Bila+, EuroSys '13]、
[Rajagopalan+, VEE '12]
[Das+, VLDB '11]
[Elmore+, SIGMOD '11]
[Cully+, '08]
など

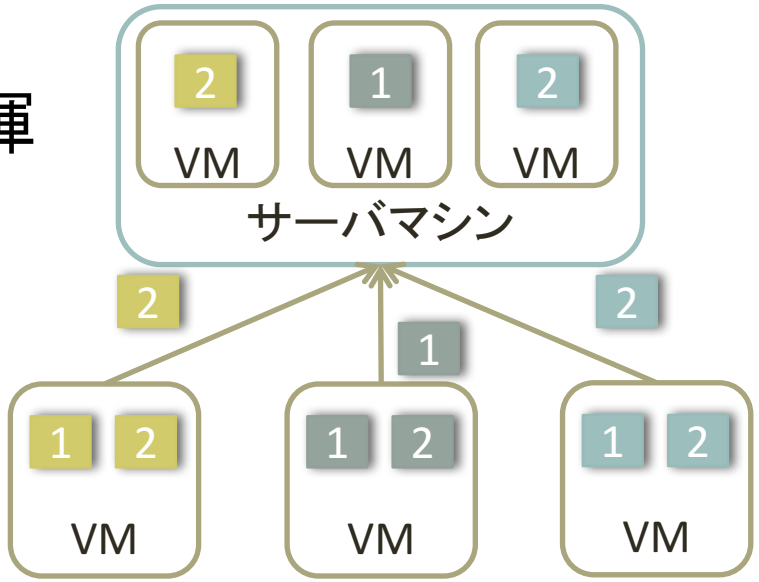
Delta Compression

- 前回と更新との差分のみを転送することで、メモリ転送量の削減を狙う [Svärd+, VEE '11]
  - 書き換え前のページと書き換えられたページの XOR を取る (Delta ページ)
    - 変化のあった部分だけビットが1になる
  - Run Length Encoding (RLE) で圧縮
    - 例: AAABBCCCCCC → A3B2C6
  - 圧縮した Delta ページを転送し、元に戻す
    - 移送先ホストは書き換え前のページを持っている
- Write-intensive なワークロードにおいて効果を発揮



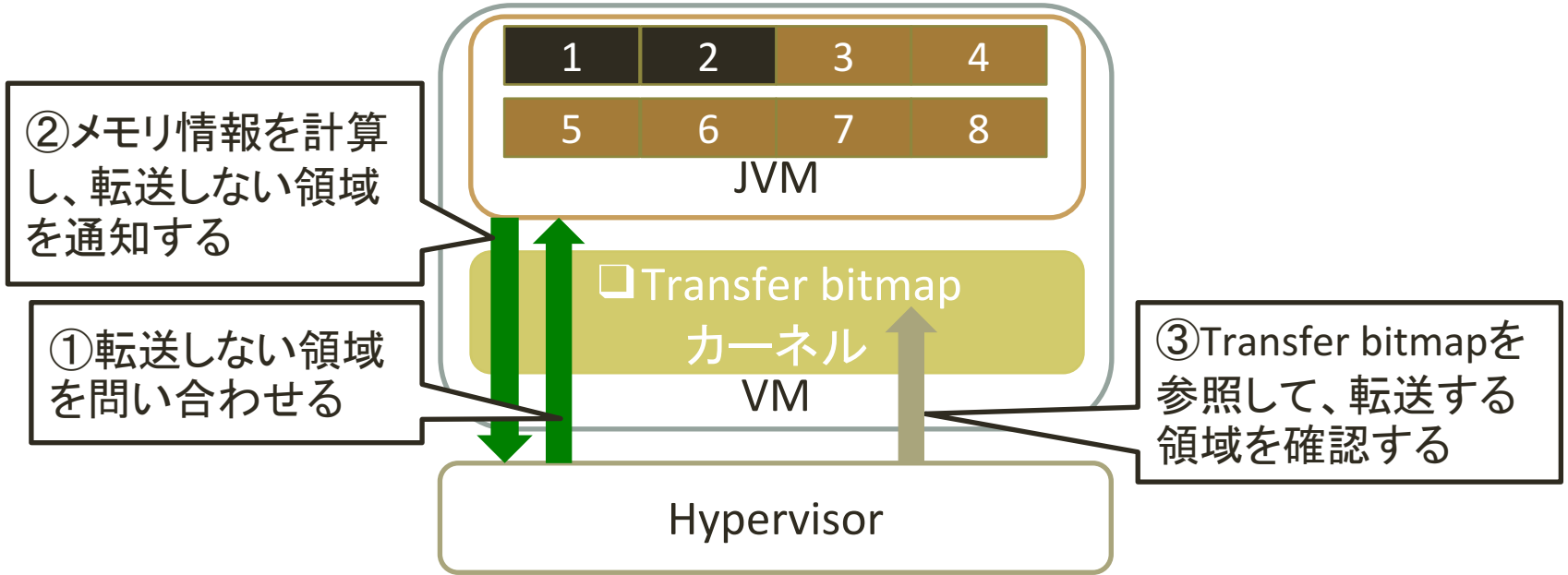
Partial Migration

- VM の一部を別マシンに移送する手法 [Bila+, EuroSys '13]
  - サーバ上の VM は稼働中のプロセスの実行に必要なデータのみを保持
    - 仮想 CPU や各種デバイスの状態
    - メモリやディスクはアクセスがあった部分を中心に保持
  - 必要なページやディスクのデータはオンデマンドで移送元マシンから転送
- オフィス環境で省電力効果を発揮
  - オフィスのマシンはほとんど Idle
    - Idle 時には 1 台のサーバマシンに集約する
    - 利用するときには移送元マシンで稼働させる



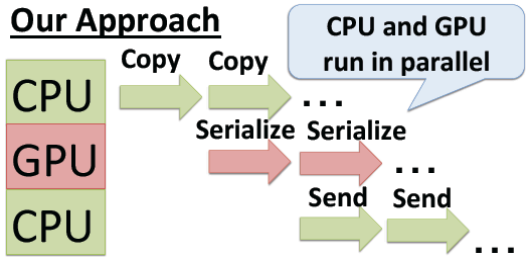
Java-Aware VM Migration

- JVM の知識を利用してメモリ転送を効率化する手法 [Hou+, EuroSys '14]
  - オブジェクトを管理する Young 領域を転送しない
    - Young 領域: 短命なオブジェクトの領域
      - 短命なオブジェクトは更新が頻繁ですぐに不要となる傾向
      - Stop & Copy Phase 時にだけ young 領域のオブジェクトを送る
  - JVM は自身のメモリ管理情報をハイパーバイザに伝える

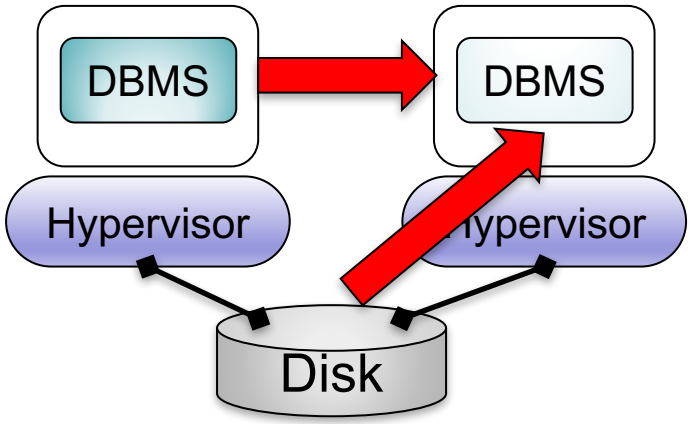


話者の取り組み

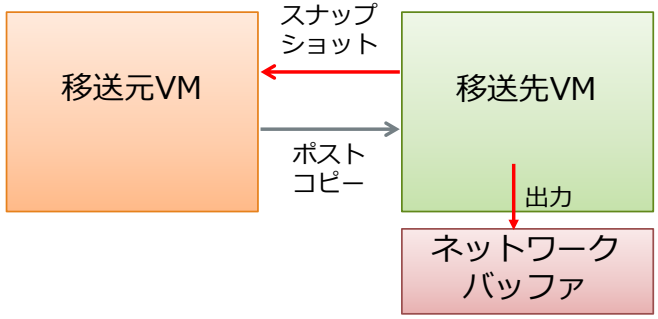
- GMigrate
  - 圧縮型ライブ VM 移送を GPGPU で アクセラレーション
  - CPU 負荷低減・移送時間の短縮に貢献



- DBMS を考慮したライブ VM 移送
  - DBMS のバッファプール構築は ストレージからのフェッチによって行なう
    - その他の領域は通常どおり行い、 VM 構築を並列化
  - 移送時間の短縮に貢献



- キャンセラブルな Post-copy
  - Stop & Copy Phase 後に 更新のあったページを移送先に転送
    - 移送元で復帰できるようにする
  - Post-copy の信頼性向上に貢献



ライブ VM 移送とアプリ

- アプリ側の力を借りると激的に円滑に実行できる
  - ライブ VM 移送の性能はアプリの挙動に大きく依存
  - データセンタを考えれば、1 VM に 1 アプリ稼動している状況を考えればよい

  ⇒ 簡単な挙動がわかるだけでも移送がしやすくなる

- 移送を実行するタイミングもシビア
  - 移送の実行はゼロコストではない
  - アプリのセマンティクスはシステム側ではわからない

  ⇒ 移送してよいタイミングを教えてもらえるだけでも運用面では非常に助かる

まとめ

- ライブ VM 移送について
  - まずは仮想化技術について簡単に説明した
  - 3 種類のメモリ転送アルゴリズム
    - Pre-copy
    - Post-copy
    - Hybrid
  - 研究成果のいくつかを紹介した
- ライブ VM 移送の研究はまだまだホット
  - なにか琴線に触れていれば幸いです
- 参考: Yamada H.,: Survey on Mechanisms for Live Virtual Machine Migration and its Improvements, JSSST Journal on Computer Software.