Optimising Cloud Computing with SBSE

David R. White & Jeremy Singer
{david.r.white, jeremy.singer}@glasgow.ac.uk
University of Glasgow

Monday 25 July 2011

Outline

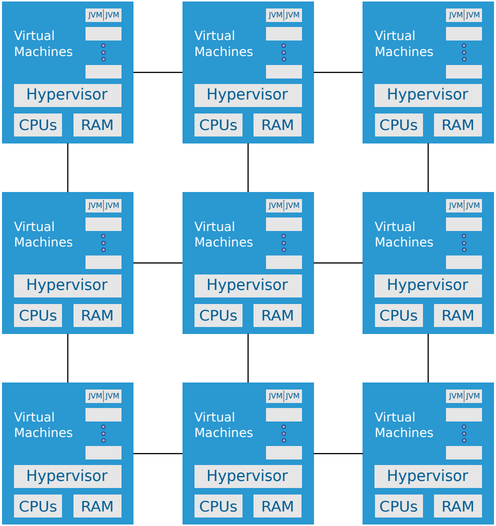
Outline

Virtual Machines

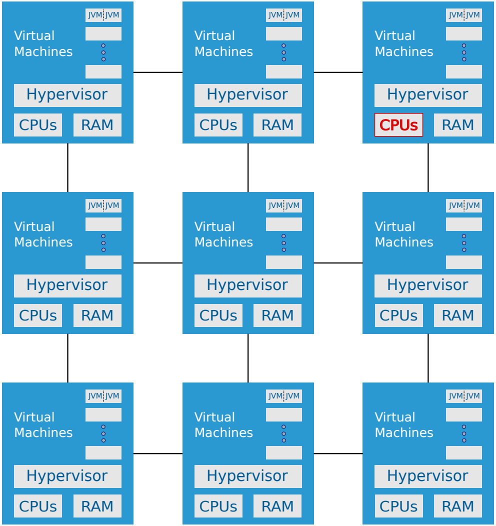
Opportunities for SBSE

Take-home

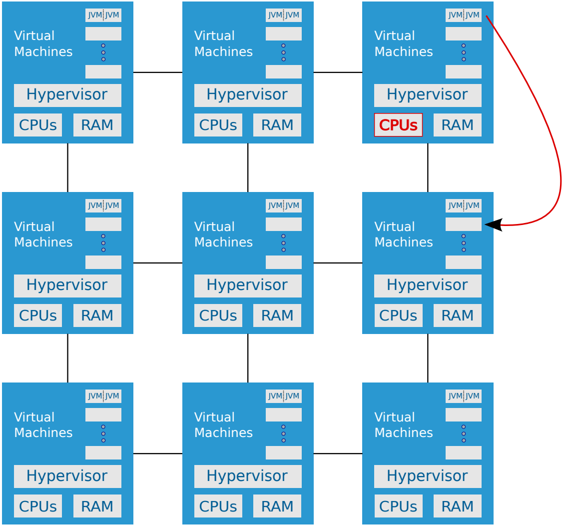
OPTIMISING VIRTUAL MACHINE MANAGEMENT



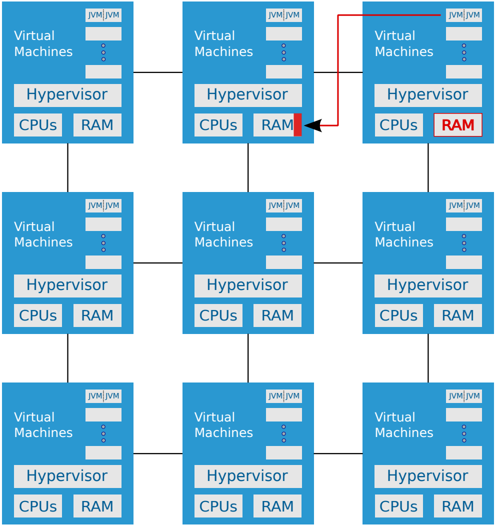
OPTIMISING VIRTUAL MACHINE MANAGEMENT



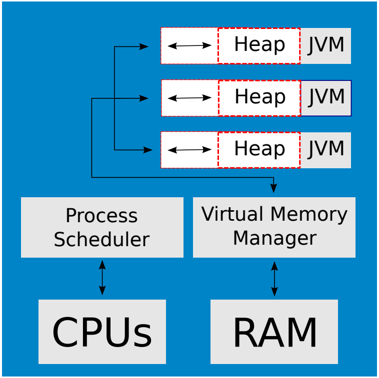
OPTIMISING VIRTUAL MACHINE MANAGEMENT



OPTIMISING VIRTUAL MACHINE MANAGEMENT



OPTIMISING A SINGLE JAVA VIRTUAL MACHINE



Competition for resources must be managed.

JVM CONFIGURATION

What heap sizes should I use?

```
-Xms<number><unit> Initial size of heap
-Xmx<number><unit> Maximum size of heap
```

JVM CONFIGURATION - II

Which Garbage Collector should I use?

- SemiSpace
- MarkSweep
- GenCopy
- GenMS
- CopyMS
- RefCount

JVM CONFIGURATION - II

Which Garbage Collector should I use?

- SemiSpace
- MarkSweep
- GenCopy
- GenMS
- CopyMS
- RefCount

Many, many more decisions to be made…

JVM CONFIGURATION: DETAIL - I

schedulingMultiplier	
eagerCompleteSweep	Should we eagerly finish sweeping at the start of a collection
protectOnRelease	Should memory be protected on release?
noFinalizer	Should finalization be disabled?
noReferenceTypes	Should reference type processing be disabled?
fullHeapSystemGC	Should a major GC be performed when a system GC is triggered?
ignoreSystemGC	Should we ignore calls to java.lang.System.gc?
variableSizeHeap	Should we shrink/grow the heap to adjust to application working set?
eagerMmapSpaces	If true, all spaces are eagerly demand zero mmapped at boot time
markSweepMarkBits	Number of bits to use for the header cycle of mark sweep spaces
stressFactor	Force a collection after this much allocation
metaDataLimit	Trigger a GC if the meta data volume grows to this limit
boundedNursery	Bound the maximum size of the nursery to this value
fixedNursery	Fix the minimum and maximum size of the nursery to this value
threads	Number of GC threads to use
enable_recompilation	Should the adaptive system recompile hot methods?
enable_precompile	Should the adaptive system precompile all methods given in the advice file before the user thread is sta
adaptive_inlining	Should we use adaptive feedback-directed inlining?
osr_promotion	Should AOS promote baseline-compiled methods to opt?
background_recompilation	Should recompilation be done on a background thread or on next invocation?
method_sample_size	How many timer ticks of method samples to take before reporting method hotness to controller
decay_frequency	After how many clock ticks should we decay
dcg_decay_rate	What factor should we decay call graph edges hotness by
dcg_sample_size	After how many timer interrupts do we update the weights in the dynamic call graph?
inline_ai_seed_multiplier	Initial edge weight of call graph is set to AI_SEED_MULTIPLER * (1/AI_CONTROL_POINT)
inline_ai_hot_callsite_threshold	What percentage of the total weight of the dcg demarcates warm/hot edges
offlinePlan	Name of offline inline plan to be read and used for inlining
early_exit_time	Value of controller clock at which AOS should exit if EARLY_EXIT is true
invocation_count_threshold	Invocation count at which a baseline compiled method should be recompiled
invocation_count_opt_level	Opt level for recompilation in invocation count based system
counter_based_sample_interval	What is the sample interval for counter-based sampling
max_opt_level	The maximum optimization level to enable.
focus_effort	Focus compilation effort based on frequency profile data
reads_kill	Should we constrain optimizations by enforcing reads-kill?
inline	Inline statically resolvable calls
inline_guarded	Guarded inlining of non-final virtual calls
inline_guarded_interfaces	Speculatively inline non-final interface calls

JVM CONFIGURATION: DETAIL - II

inline_preex	Pre-existence based inlining
simplify_integer_ops	Simplify operations on integers
simplify_long_ops	Simplify operations on longs
simplify_float_ops	Simplify operations on floats
simplify_double_ops	Simplify operations on floats
simplify_ref_ops	Simplify operations on references
simplify_tib_ops	Simplify operations on TIBs
simplify_field_ops	Simplify operations on fields
simplify_chase_final_fields	Chase final fields avoiding loads at runtime
local_constant_prop	Perform local constant propagation
local_copy_prop	Perform local copy propagation
local_cse	Perform local common subexpression elimination
local_expression_folding	Should we try to fold expressions with constants locally?
control_static_splitting	CFG splitting to create hot traces based on static heuristics
control_unwhile	Turn whiles into untils
escape_simple_ipa	Eagerly compute method summaries for simple escape analysis
escape_scalar_replace_aggregates	If possible turn aggregates (objects) into variable definition/uses
escape_monitor_removal	Try to remove unnecessary monitor operations
escape_invokee_thread_local	Compile the method assuming the invokee is thread-local. Cannot be properly set on command line.
ssa	Should SSA form be constructed on the HIR?
ssa_expression_folding	Should we try to fold expressions with constants in SSA form?
ssa_redundant_branch_elimination	Eliminate redundant conditional branches
ssa_licm_ignore_pei	Assume PEIs do not throw or state is not observable
ssa_load_elimination	Should we perform redundant load elimination during SSA pass?
ssa_coalesce_after	Should we coalesce move instructions after leaving SSA?
ssa_loop_versioning	Create copies of loops where runtime exceptions are checked prior to entry
ssa_live_range_splitting	Split live ranges using LIR SSA pass?
ssa_gcp	Perform global code placement
ssa_gcse	Perform global code placement
ssa_global_bounds	Perform (incomplete/unsafe) global Array Bound Check elimination on Demand
ssa_splitblock_to_avoid_rename	When leaving SSA create blocks to avoid renaming variables
ssa_splitblock_for_local_live	When leaving SSA create blocks for local liveness
ssa_splitblock_into_infrequent	When leaving SSA create blocks to avoid adding code to frequently executed blocks
reorder_code	Reorder basic blocks for improved locality and branch prediction
reorder_code_ph	Reorder basic blocks using Pettis and Hansen Algo2
h2l_inline_new	Inline allocation of scalars and arrays
h2l_inline_write_barrier	Inline write barriers for generational collectors
h2l_inline_primitive_write_barrier	Inline primitive write barriers for certain collectors
h2l_no_callee_exceptions	Assert that any callee of this compiled method will not throw exceptions. Cannot be properly set on comm
h2l_call_via_jtoc	Plant virtual calls via the JTOC rather than from the tib of anobject when possible
l2m_handler_liveness	Store liveness for handlers to improve dependence graph at PEIs
l2m_schedule_prepass	Perform prepass instruction scheduling

JVM CONFIGURATION: DETAIL - III

```
regalloc_coalesce_moves             Attempt to coalesce to eliminate register moves?
regalloc_coalesce_spills            Attempt to coalesce stack locations?
adaptive_instrumentation_sampling   Perform code transformation to sample instrumentation code.
adaptive_no_duplication             When performing inst. sampling, should it be done without duplicating code?
adaptive_processor_specific_counter Should there be one CBS counter per processor for SMP performance?
adaptive_remove_yp_from_checking    Should yieldpoints be removed from the checking code (requires finite sample interval)
osr_guarded_inlining                Insert OSR point at off branch of guarded inlining?
osr_inline_policy                   Use OSR knowledge to drive more aggressive inlining?

profile_edge_count_input_file       Input file of edge counter profile data
profile_infrequent_threshold        Cumulative threshold which defines the set of infrequent basic blocks
profile_cbs_hotness                 Threshold at which a conditional branch is considered to be skewed
escape_max_array_size               Maximum size of array to replaced with registers by simple escape analysis
ssa_load_elimination_rounds         How many rounds of redundant load elimination will we attempt?
l2m_max_block_size                  Maximum size of block for BURS, larger blocks are split
regalloc_simple_spill_cost_move_factor             spill penalty for move instructions
regalloc_simple_spill_cost_memory_operand_factor   spill penalty for registers used in memory operands
control_tableswitch_cutoff          If a tableswitch comprises this many or fewer comparisons convert it into multiple if-then-else style br
control_cond_move_cutoff            How many extra instructions will we insert in order to remove a conditional branch?
control_unroll_log                  Unroll loops. Duplicates the loop body 2^n times.
control_static_splitting_max_cost   Upper bound on the number of instructions duplicated per block when trying to create hot traces with sta
control_well_predicted_cutoff       Don't replace branches with conditional moves if they are outside of the range of 0.5 +- this value
inline_max_target_size              Static inlining heuristic: Upper bound on callee size
inline_max_inline_depth             Static inlining heuristic: Upper bound on depth of inlining
inline_max_always_inline_target_size Static inlining heuristic: Always inline callees of this size or smaller
inline_massive_method_size           Static inlining heuristic: If root method is already this big, then only inline trivial methods
inline_max_arg_bonus                Maximum bonus for reducing the perceived size of a method during inlining.
inline_precise_reg_array_arg_bonus      Bonus given to inlining methods that are passed a register of a known precise type.
inline_declared_aastored_array_arg_bonus Bonus given when there's potential to optimize checkstore portion of aastore bytecode on parameter
inline_precise_reg_class_arg_bonus  Bonus given to inlining methods that are passed a register of a known precise type.
inline_extant_reg_class_arg_bonus   Bonus given to inlining methods that are passed a register that's known not to be null.
inline_int_const_arg_bonus          Bonus given to inlining methods that are passed an int constant argument
inline_null_const_arg_bonus         Bonus given to inlining methods that are passed a null constant argument
inline_object_const_arg_bonus       Bonus given to inlining methods that are passed an object constant argument
inline_call_depth_cost              As we inline deeper nested methods what cost (or bonus) do we wish to give to deter (or encourage) nesti
inline_ai_max_target_size           Adaptive inlining heuristic: Upper bound on callee size
inline_ai_min_callsite_fraction      Adaptive inlining heuristc: Minimum fraction of callsite distribution for guarded inlining of a callee
```

Outline

WHY SBSE?

Manual optimisation is difficult: we have a complex system

Variables include software's behaviour, phase, and interactions

Solutions are non-obvious and require creativity.

A COMMERCIALLY RELEVANT PROBLEM FOR GP

"I think GP has a toy problem problem."

Sean Luke, June 2011.

This is most certainly *not* a toy problem!

Existing Optimisations

How are decisions made at the moment?

Existing Optimisations

How are decisions made at the moment?

Answer: they're often not - just deferred!

Existing Optimisations

How are decisions made at the moment?

Answer: they're often not - just deferred!

Most recent version of Hotspot JVM has some adaptation.

Existing Optimisations

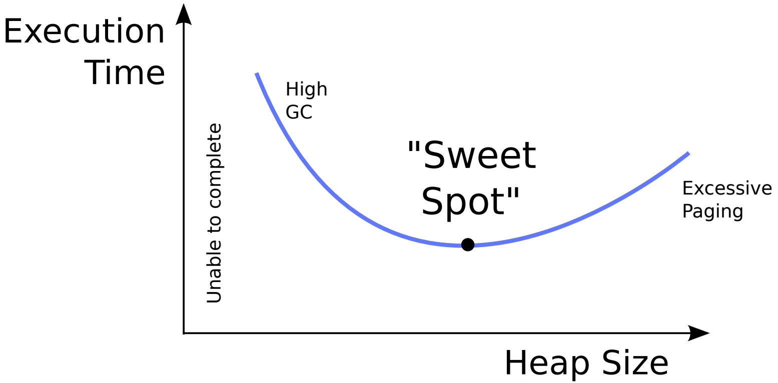
How are decisions made at the moment?

Answer: they're often not - just deferred!

Most recent version of Hotspot JVM has some adaptation.

This is a case of "Best effort" - so why not GP?

A CONCRETE EXAMPLE: HEAP SIZE CONTROL



JIKES RVM HEAP RESIZING

GC Overhead \ Heap Occupancy	0.00	0.10	0.30	0.60	0.80	1.00
0.00	0.90	0.90	0.95	1.00	1.00	1.00
0.01	0.90	0.90	0.95	1.00	1.00	1.00
0.02	0.95	0.95	1.00	1.00	1.00	1.00
0.07	1.00	1.00	1.10	1.15	1.20	1.20
0.15	1.00	1.00	1.20	1.25	1.35	1.30
0.40	1.00	1.00	1.25	1.30	1.50	1.50
1.00	1.00	1.00	1.25	1.30	1.50	1.50

Look-up table for heap-resize coefficient.

DESIGN DECISIONS

(Private Communication)

> *"…back in 2003 [anon] and I did some experimental tuning and came up with the numbers by eyeballing things. At the time, it seemed to be somewhat stable and making reasonable decisions but that was also about 4 major versions ago and I don't think anyone has really looked at it seriously since then. I think there was some amount of sensitivity to the values…"*

Outline

Virtual Machines

Opportunities for SBSE

Take-home

SUMMARY

- Cloud infrastructure design presents new software engineering challenges.
- Scheduling and memory management are open to optimisation.
- Many problems are amenable to SBSE.
- We are looking for collaborators!

...and relevant existing work!

{david.r.white, jeremy.singer}@glasgow.ac.uk

SEE ALSO

Cloud computing: state-of-the-art and research challenges.
Zhang et al.
Journal of Internet Services and Applications, 1/1, 2010. pp. 7-18.

Overdriver: handling memory overload in an oversubscribed cloud.
Williams et. al.
Proceedings of VEE 2011.

Previous work on service (application) level by Wada et al. and Nallur et al.

–

{david.r.white, jeremy.singer}@glasgow.ac.uk