Combining Moore’s Law, Amdahl’s Law, and Communication
How Software Can Save Moore’s Price/Performance Model

Kevin David Howard, Chief Technology Officer, MPT, Inc., Kevin@massivelyparallel.com
James A. Lupo, Senior Research Scientist, MPT, Inc., James@massivelyparallel.com

Extended Abstract

Combining Moore’s law, which describes hardware costs/performance, and Amdahl’s law, which describes algorithm to multiple chip speed-up effects, with new communication models is shown to maintain or enhance Moore’s model.

Moore’s law is usually interpreted to mean “the number of transistors that can fit onto a square inch of silicon doubles every 12 months.”[1,2] However, Moore’s statement covered much more than the density doubling discussed so often. A new perspective is important if we are to be able to extend the law’s life using software. Moore stated:

> “The complexity for minimum component costs has increased at a rate of roughly a factor of two per year.”[2]

Rather than a statement of raw density, it is a statement of the most cost effective chip manufacturing density, taking into consideration four factors: 1) the maximum number of transistors per square inch, 2) the size of the wafer, 3) the average number of defects per square inch, and 4) the costs associated with integrating multiple components (interconnection, packaging, PCB, other multi-chip integration costs).[3] A valid interpretation of Moore’s principal, showing Moore’s results, is given by the following equation:

$$\mathbf{C_U = C_I(n_c) + f(n_f)\ (C_w\ n_c\ n_f\ A_f)\ /\ (P(n_f)\ A_w)}$$

Where: C_U = total cost per unit
C_I = chip integration cost per unit
C_w = wafer fabrication cost
n_c = # of chips per unit
n_f = # of features per chip
A_f = average feature area
$P(n_f)$ = % yield for chips with n_f features
A_w = size of wafer (area)
$f(n_f)$ = percentage of A_w covered by chips

Further, cost effective performance is related to the number of features that can be placed on a chip, the chip clock rate, and the energy consumed by the chip. We will extend the idea of the number of features per chip to include the concept that a feature is a structure that can transform a bit pattern. Since a mathematical algorithm performs a transform using the chip features, we can associate the idea of transforms per unit time with algorithms.

If we fix the performance, then increasing the number of transforms per unit time requires an increase in the number of chips (processors) used. Using multiple processors requires the use of Amdahl’s law.

Amdahl's law is the standard way of describing parallel speed-up.[4] By deriving this law directly from parallel communication and parallel information processing theory, we show the roles played by communication channels, communication topology, and processor speeds in recasting the interpretation of the terms of the law. This interpretation requires the coupling of communication and computation effects. We show the communication overhead to be:

$$\mathbf{\Omega = t_{\lambda 1} + t'_1 + \Sigma\ \max((t'_{i,i+1} - t^l_{i+1}),0) + t_d}$$

Then Amdahl's law can be expressed as:

$$\mathbf{S(P) = (\Omega_{single} + t_p)/[\Omega_{multi} + t_{\ c} + T(t_p,P)/P]}$$

Where: $S(P)$ ≡ Speed-up of P processors (performance).
Ω_{single} ≡ Single processor I/O overhead.
Ω_{multi} ≡ Multi-processor I/O overhead.
$t_{\lambda 1}$ ≡ initial communication latency.
t_c ≡ cross-communication data transfer time for current model
t'_1 ≡ initial data priming time.
t_d ≡ data draining time.
$t'_{i,i+1}$ ≡ cross communication data priming times.
t_p ≡ processing time on single processor.
$T(t_p,P)$ ≡ processing time on P processors.
P ≡ number of processors.
t^l_{i+1} ≡ lead time from start of overlapped exchange to end of processing step i.

Combining this interpretation with several new communication models, we demonstrate how software can be used to extend the life of Moore’s law. **Figure 1** is an example of a logical 80-processor scatter/gather model.[5] The bi-section bandwidth = 20 x the individual processor port bandwidths. **Figure 2** shows the end-to-end effect of this new communication model on an algorithm using 80 processors in parallel. It becomes clear that the term software includes the communication model used.

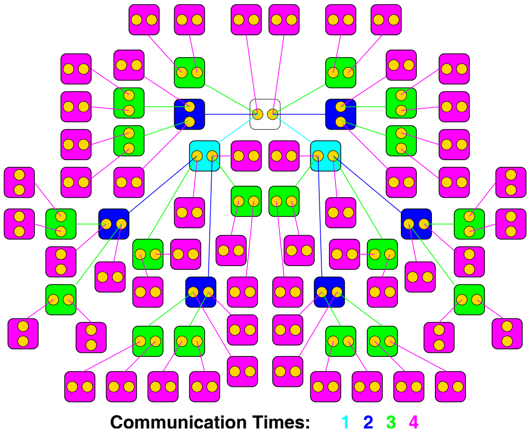

Figure 1: 80-Node Communication Model

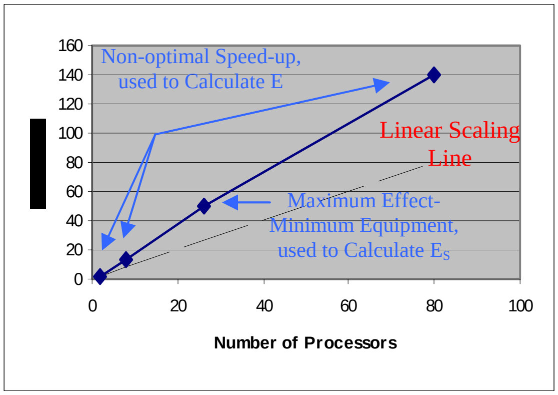

Figure 2: 20,000 Digits of e^x Observed Performance

Moore's law can be expressed in terms of operations per second by creating a function O(n_f,*a*) which defines the number of operations per second achieved by a processor with n_f features on some algorithm *a*. Though this cost analysis applies to one processor, we can extend Moore's law to multi-processor systems with the aid of Amdahl's Law. We define the integration cost of a multi-processor system and the total system cost as:

$$\mathbf{C_I(P) = PN(C_{nic} + C_{fabric})}$$
$$\mathbf{C_s(P) = PC_U + C_I(P)}$$

Where: $C_I(P)$ ≡ parallel integration cost
$C_S(P)$ ≡ total system cost
P ≡ number of processors
N ≡ number of channels per processor
C_{nic} ≡ interface cost per channel
C_{fabric} ≡ network fabric cost per processor

The effective cost performance factor for parallel systems then takes the form:

$$\mathbf{E = S(P)\ ((PC_U + C_I(P))/(O(n_f,\mathit{a})P)}$$

We can then define the optimized system cost $\mathbf{E_S}$ **as the minimum of E w.r.t. P**. This is the minimum cost needed to achieve a performance level, given the hardware and software characteristics of a system and the algorithm it is operating on. **Figure 3** shows the curves of five different communication models. Each model generates a different speed-up effect (sub-linear, partial linear, linear, partial super-linear, and super-linear). Thus, the cost/performance curve generated by each effect is shown, with the standard Moore's law cost/performance curve for comparison.

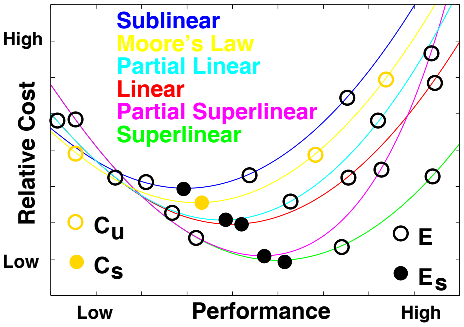

Figure 3: Modified Moore Cost Performance Chart. Each Color Represents a Different Communication Model

Any adjustment that moves the cost curve to the right and/or down, as shown by E_S, represents an effective hardware performance improvement. Finding C_s and E_s for several algorithms allows the generation of a Kiviat diagram, **Figure 4**, that can be used to represent the total system cost/performance.

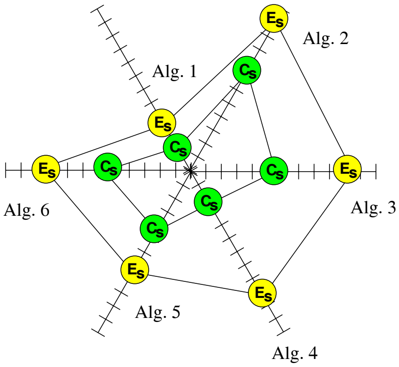

Figure 4: System Cost/Performance Kiviat Diagram

[1] http://www.intel.com/technology/silicon/mooreslaw/
[2] Moore, G., Electronics, Vol. 38, Number 8, April 19, 1968
[3] http://news.com.com/New+life+for+Moores +Law/2009-1006_3-5672485.html?tag=nl
[4] http://research.microsoft.com/users/ GBell/Computer_Structures_ Principles_and_Examples/csp0322.htm
[5] Patent pending