Ternary CAM Power and Delay Model: Extensions and Uses

Banit Agrawal, *Student Member, IEEE*, and Timothy Sherwood, *Member, IEEE*

***Abstract*—Applications in computer networks often require high throughput access to large data structures for lookup and classification. While advanced algorithms exist to speed these search primitives on network processors and even custom application-specific integrated circuits (ASICs), achieving tight bounds on worst case performance with standard memories often requires a very careful analysis of all possible access patterns. An alternative, and often times more simple solution, is possible if a ternary CAM (TCAM) is used to perform a fully parallel search across the entire data set. Unfortunately, this parallelism means that large portions of the chip are switching during each cycle, causing large amounts of power to be consumed. While researchers at all levels of design (from algorithms to circuits) have begun to explore new ways of managing the power consumption, quantifying design alternatives is difficult due to a lack of available models. In this paper, we examine the structure of a modern TCAM and present a simple, yet accurate, power and delay model. We present techniques to estimate the dynamic power consumption and leakage power of a TCAM structure and validate the model using a combination of industrial TCAM datasheets and prior published works. Such a model is a critical first step in bridging the intellectual divide between circuit-level and algorithm-level optimizations. To demonstrate the utility of our model, we present an extensive analysis of the model by varying various architectural parameters and describe how our model can be easily extended to handle several circuit optimizations in the TCAM structure. In addition, we present a comparative study of SRAM and TCAM energy consumption to directly quantify the many design options which will be very useful for network designers to explore various power management schemes.**

***Index Terms*—Content addressable memory (CAM), delay, dynamic, leakage, model, network algorithms, power, router, SRAM, ternary CAM (TCAM).**

I. Introduction

HIGH SPEED search is one of the most fundamental duties of a network device. Whether or not it is a classic problem, such as Internet Protocol (IP)-lookup or an emerging domain such as worm detection, the ability to index and search large amounts of state with incredibly high throughput is critical to a variety of important network algorithms. For example, in the case of IP-lookup, the state is a large routing table filled with prefixes, while for worm detection the state may be a set of patterns to match. While there is a great deal of work on advanced algorithms to speed the search through this state with traditional memories, the complexities of implementation motivate many system developers to use specialized memory designs that directly support search as an access primitive. In particular, many systems use content addressable memories (CAMs) to provide the required search capabilities with a minimum of additional cost and complexity.

CAMs and, specifically, ternary CAMs (TCAMs) are used extensively in networking. CAMs provide read and write, such as a normal memory, but additionally support *search* which will find the index of any matching data in the *entire* memory. A TCAM extends this functionality to include wildcard bits that will match both one and zero. These wildcards can be used on both the access operations of the memory (indicating some bits of the search are "don't cares") or can be stored with the data itself (indicating some bits of the data should not be used for determining a match). The fully parallel search provided by TCAM eases the implementation of many complex operations such as routing table lookup. Due to the fact that the TCAM searches every location in memory at once, the ordering of the elements in the TCAM is less important and large indexing structures can often times be entirely avoided. This parallel search directly implements the requirements of some applications (such as, IP-lookup [1]–[4]) and can serve as the building block of more complex searching schemes [5]. TCAM also finds applications in other high-speed network operations such as packet classification [3], access list control, and pattern matching for intrusion detection [6]. Indeed, recently it has been proposed that TCAM could even be used outside the networking domain to accelerate various database search primitives [7], [8].

While there are many advantages of using TCAM, a fully parallel search of memory does not come for free. In the power constrained situations that most high performance routers find themselves, these searches can, if unoptimized, easily consume tens or hundreds of Watts. To give an example of growing power concerns, Cisco Systems provides a 600-W redundant AC power system, which can support up to four 150-W external network devices such as routers [9]. As TCAMs are increasingly integral components in next-generation routers and network search engines (NSEs) [10], they are a natural target for optimization. While previous work has addressed the issue of power consumption in network processors [11], there is no work that accounts for TCAM power consumption. There are complex tradeoffs at work, the choice of algorithms, the amount of work offloaded to the TCAM, and the structure and size of the TCAM itself, all impact one another. However, in order for the network community to begin to address these problems, a simple yet accurate TCAM power model is required.

The traditional computer architecture community has been well served by the adoption of Cacti [12], eCacti [13], and other

Manuscript received February 24, 2007; revised June 25, 2007. This work was supported in part by the National Science Foundation (NSF) under NSF Career Grant CCF-0448654 and NSF-CT-T 0524771. This work is an extended version of the authors' work in the Proceedings of ISPASS 2006, pp. 120-129.

The authors are with the Computer Science Department, University of California, Santa Barbara, CA 93106 USA (e-mail: banit@cs.ucsb.edu; sherwood@cs.ucsb.edu).

Digital Object Identifier 10.1109/TVLSI.2008.917538

architecture level memory models. These models help designers evaluate various on-chip cache designs and have led to a variety of different research endeavors that seek to make tradeoffs across the traditional boundaries of architecture and circuits. In this paper we present the first publicly available power model for TCAM with the intent of clarifying and encapsulating the most important aspects of the design with the intent that it makes it possible to quantify the impact of novel cross-layer optimizations.

While many different TCAM designs have been proposed [14]–[17], most modern designs share a similar XNOR based cell design. We provide a description of TCAM operation as it relates to power modeling in Section III. Building on this understanding of the TCAM internals, in Section IV we abstract away many of the less important aspects and converge on a simple to use model, based on simple but effective approximations of the line capacitances and switching activities. One of our main contributions is a model that is more accurate and useful than a simple watts/bit approximation. If the architecture and networking communities are to develop new algorithms and TCAM power management schemes, we must embrace a model that exposes the *opportunities for improvement*. This means that we must consider the effects of different length TCAM entries on power consumption, the banking of TCAMs to reduce word and match line capacitance, and the effect of the priority encoder. In Section V, we present our model and show that it matches closely with recently published circuit data for a variety of configurations. We also show how the energy consumption of TCAM changes based on the architectural parameters. We also present the leakage power analysis, delay analysis, and show how our model can be simply extended to account for some circuit-level optimizations. To demonstrate how our model can be useful to the network designers, we also present a comparative study of SRAM and TCAM energy consumption as an example of how design alternatives and power management schemes can be directly evaluated.

II. Related Work

Although TCAMs are very useful in high-speed networking applications, many network designers worry about power dissipation/consumption. In fact, in recent years a number of techniques have been proposed to reduce TCAM power consumption [1], [2], [18] by searching in only a subset of the TCAM. While these optimizations may prove useful, it has been impossible for designers to quantify the effectiveness of their approaches due to a lack of power and delay models. For example, in CoolCAM [1] the authors assume that the power consumption is simply proportional to the number of rows. They provide a set of clever algorithms and a two-level TCAM design which requires searching less rows, which in turn reduces the total power consumption for search operation. Although we find their assumptions to be a good approximation for making relative estimations, an absolute quantitative figure for power saving in terms of joules or watts would be far more valuable. In EaseCAM [2], a page based scheme is used to reduce the power consumption and they base their savings on the CAM implementation inside Cacti (which is different than a TCAM). In addition to these row pruning techniques, there is also some work which proposes to reduce the number of bits of comparison [19]. Therefore, we need a power model for TCAM which can also take column bits as the input parameters to estimate the dynamic power.

While we are the first to provide a usable TCAM power model, there are many power modeling tools for other structures such as Orion [20], Wattch [21], Cacti [22], which model power consumption in different ways. Hsiao *et al.* [23] provide a power model for CAM which accounts for evaluation power, input transition power and clock power. Wang *et al.* [11] provide a power model for on-chip routers that model various components of routers such as first-input–first-outputs (FIFOs), crossbar, and memories including SRAM and CAM. The standard CAM cell is quite different than the TCAM cell as TCAM cell usually requires two storage cells for one bit and additional circuit components, and these models do not account for TCAM cell and the sizing of various transistors in TCAM. We have found no previous work that provides a publicly available power modeling tool for TCAM, yet there is a pressing need from the networking community to quantify the power savings for various TCAM power management schemes. We provide a publicly available modeling tool for TCAM, which can take high-level architectural parameters as input and provides an estimate of dynamic power per access as output.

As technology modeling is always like trying to match a moving target, our model is scalable with respect to CMOS technology as it takes both CMOS feature size and wire parameters from the ITRS roadmap [24] and TCAM cell layout parameters as input parameters. Hence, we can estimate to first order the dynamic power of any TCAM design by taking care of the TCAM cell used. In addition, our model takes architectural parameters such as number of rows, number of search bits, and number of banks as input parameters.

III. TCAM Structures

To ground our modeling technique, it is worth describing the internal structure of a modern TCAM design, along with the various design options and parameters. In the following subsection, we describe the read, write, and search operations of a high speed TCAM, and describe the primary channels of power consumption to motivate our model.

A. TCAM Architecture

Before we present the details of our model, it is worth reviewing the important structures in a TCAM, particularly as they relate to power consumption. In particular, we concentrate primarily on the aspects relating to search as this dominates the overall system power on an active device. The fundamental operations of a TCAM are:

- *write*—updates the entries in a row of TCAM cells;
- *read*—reads the contents of a row of TCAM cells;
- *search*—finds a match across all TCAM rows.

Fig. 1 shows the generic design of a XNOR-based TCAM. At a high level (pictured on the left) the TCAM has three major components: the precharge unit, the actual array of TCAM cells, and the priority encoder. The precharge unit is needed to precharge the match lines before a search. During a search, all of the bits in the TCAM cells are compared against the bits driven on the select line. After the search is complete, the rows that match, will have their match line set high, and all others will be set low. Because there is the potential for multiple matches on a given search, especially when searches and data include “don’t care”

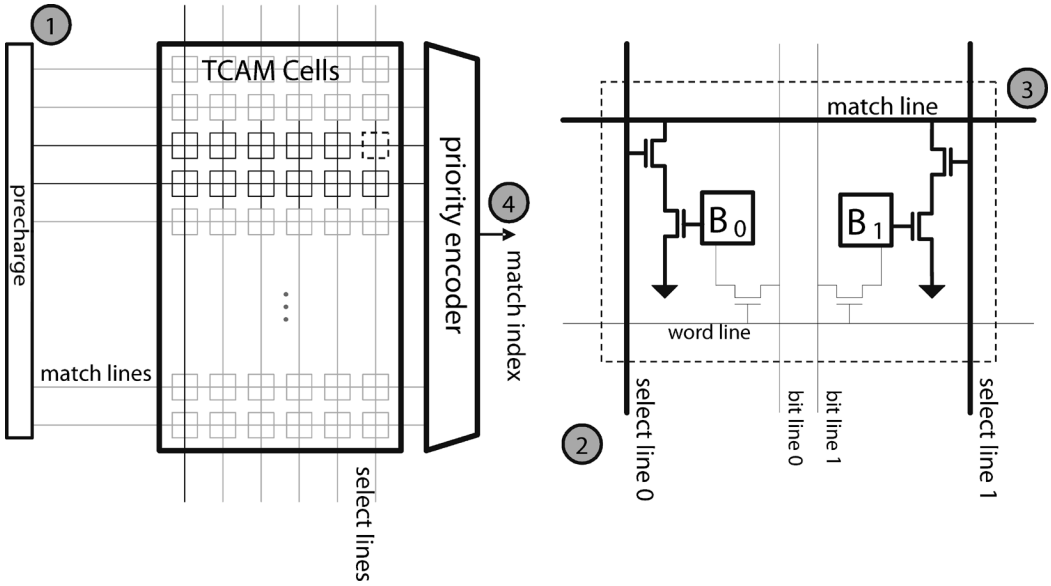

Fig. 1. Conventional TCAM architecture along with TCAM cell structure is shown. The major components of a TCAM architecture are the precharge circuit, the TCAM cells array, and the priority encoder as shown on the left hand side. On the right-hand side, a typical TCAM cell structure with *matchlines* and *searchlines* is shown. The bit B_0 and B_1 can be static or dynamic storage cells. The grey circles with a number show the sequence of operations in a TCAM search: 1) precharging match lines; 2) driving search lines; 3) match lines discharging on mismatch; and 4) priority encoding.

bits, a final step is needed to pick one of R matches which is the highest priority match. Typically this is implemented by keeping the TCAM in partially sorted order by priority, and using a priority encoder to select the first row that matches.

Central to the strength of Ternary CAM over a traditional CAM is the ability to include "don't care" bits in both the search and in the data. This wildcard matching enabled by TCAM is a natural fit to many applications including the prefix matching problem from IP-lookup [1], sorting [7], or range queries for packet classification [3], [5]. A "don't care" bit in the search pattern requests that a particular column of bits not be taken into consideration for determining which rows are the match. This can be used to handle data of variable length or find all entries with a common prefix. The ternary part of TCAM comes from the fact that the cell itself can encode a "don't care" bit, and thus can be in one of the three states: "match 0," "match 1," and "match both." Typically this is stored in the TCAM as two bits as shown in the right half of Fig. 1.

The biggest benefit of using a TCAM is that the comparison happens directly in the cells, which means that to understand the power consumed by the comparison operation we must understand the internals of a TCAM cell. The right side of Fig. 1 shows the design of a TCAM cell with the actual bit storage of the two states bits abstracted away as B_0 and B_1. The bold lines show the paths relevant to the search operation. A logical zero is stored in the TCAM cell when $B_0 = 1$ and $B_1 = 0$, while a logical one is stored when $B_0 = 0$ and $B_1 = 1$. The position of the "1" in either B_0 or B_1 determines where the select line comparison should occur. A logical "don't care" is stored by insuring that *no* comparison is done for this bit which is achieved with $B_0 = 0$ and $B_1 = 0$.

The cell B_0 and B_1 can be either static TCAM cells (four CMOS transistors) or can be dynamic TCAM cells (one CMOS transisor). Later, we show that our model calculates the dynamic power by taking TCAM cell layout into consideration. Hence, we are able to measure the dynamic power of any TCAM design by using its cell layout parameters.

B. Search Operation Energy

Once the TCAM cells have been set to one of the three legal states, a search can be done. The four steps shown in Fig. 1 as grey circles are as follows. 1) Before the search occurs, all of the select lines are set low to insure that the match line is insulated from ground. The match lines are then precharged, meaning that the line is set high but then disconnected from power so that the value of the line is essentially stored in the capacitance of the wire. 2) Once precharged, the select lines are driven to force the comparison to take place. To search for a logical one $Select_0$ is driven high, while a logical zero match can be found by driving $Select_1$ high. To do a "don't care" search, neither of the $Select$ lines for a given bit are used. If there is a mismatch, then the select line will be high, and the bit in one cell will be high (turning on both the search transistors) resulting in a path from the match line to ground. Thus, the charge stored on the match line will remain intact *only* if there are no mismatches. 3) In this way, the match line is acting as a very long wired NOR gate, combining the local match results in each cell to effect the status of the match line. Once the proper match lines have drained, there may still be multiple entries that match the query. Arbitration between these matches is often performed by taking the *first* match in a specified order, letting the position of the entry enforce the priority. 4) In the case of IP-lookup, this corresponds nicely to longest prefix matching as long as the entries are inserted into the TCAM in partially sorted order. The priority encoder performs this function across all of the match lines in the design.

The power consumption in the design comes primarily from the combined effect of steps 1) and 3). Every match lines in the system is filled with charge and then dumped to ground on every access. The total capacitance of these lines, combined with the

operating voltage of the match lines, determine the first order power consumption of the system. The other significant components are the toggling of the select lines (2), and the priority encoder switching (4). In this paper, we precisely quantify the access energy from each of these parts so as to provide an easy to use, general purpose, TCAM power model.

IV. Modeling of TCAM

In Section III, we described at a high level the internal TCAM architecture. It is now time to explain the power modeling approach we have taken, and derive the most important parameters. While there is some low-level circuits discussion, our end result will be a simple to use and intuitive model that has been validated against real hardware designs from which the architecture and networking communities are able to build.

The power consumption of any hardware component is mainly dependent on the voltage supply $(V_{\rm dd})$, equivalent capacitance $(C_{\rm eq})$ and the operating frequency (f). Most of this power consumption in TCAM is due to charging and discharging of various control lines. The operating frequency decides how fast these lines are charged or discharged. To find the frequency of TCAM, we need to find the access time of various components in TCAM which we describe in Section IV-D.

The worst case power consumption will be the same as the average case if on every cycle the line is toggled, switching from "0" to "1" or "1" to "0." While an estimate based on this idea will provide a bound on the energy consumption, most real designs do not toggle every line every cycle. If the capacitance of a hardware component does not charge or discharge every cycle, then we need to find out the total switching activity over a period of time to calculate the actual dynamic power. Taking activity factor into account, the dependencies for dynamic power is shown in

$$\begin{aligned} \text{dyn-power} &= C_{\rm eq} * V_{\rm dd}^2 * \text{Activity} * f \\ \text{Activity} &= \text{average switching factor}(\leq 1). \end{aligned} \quad (1)$$

We have already identified the hardware components that are responsible for the bulk of the power consumption in a search operation in Section III. We model each of these hardware components separately and calculate the equivalent capacitance $(C_{\rm eq})$. While at first glance, the switching activity may seem dependent on the actual set of IP address traces, 5-tuple packet classification rules, or input patterns for matching, in reality it is mostly independent of all these factors as we will describe. Initially, we provide the worst-case power consumption by assuming Activity is 100%, but we then relax this by examining a more realistic worst-case operating behavior.

In the next subsections we explain the power modeling of search operation, read/write operation, along with leakage power modeling and the delay modeling of TCAM.

A. Modeling Search Power

As described earlier, the dynamic power consumption in a search operation is mainly from match lines, search lines, and priority encoder. Next, we describe the power modeling of each of these components.

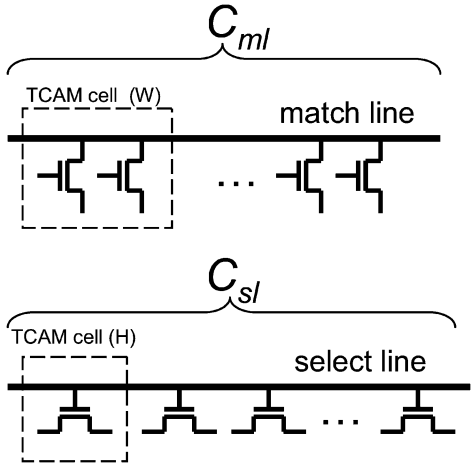

Fig. 2. Capacitive loading effect is shown for matchlines and searchlines. C_{ml} gets some load capacitance from the drain capacitance of the comparison transistors, whereas C_{sl} gets some capacitive loading effect from the gate capacitance of one comparison transistor.

1) Match Line Power: The largest amount of energy dissipated by TCAM comes from the match line dissipation. Each and every access (which could be once a cycle), all of the match lines are charged and then all but a few are discharged leaving only the matches. The amount of energy required to perform this operation is a direct function of the total capacitance of all the match lines, as that determines the amount of charge required to precharge the line to the desired voltage. If we look back to Fig. 2, we can see that the match lines run the entire length of a row, and that hanging off of the line are many transistors each of which is a potential path to ground (if a mismatch occurs). If we remove all of the components not directly connected to the match line, we end up with the top half of Fig. 2.

The capacitance of the match line is a function of both the length of the wire, and the number of transistors which source the line. For each cell in a row, there are two such transistors, one for each select line. Assuming the width of the TCAM is known, we can calculate the capacitance of a match line over l bits as shown in (2) (see Fig. 3).

Equation (2) (see Fig. 3) will estimate the capacitance of a single match line. We calculate the capacitances of the components ($C_{\rm metal-wire-per-cell-h}$, $C_{\rm drain-compareT}$, $C_{\rm nandinv-gate}$, $C_{\rm precharge}$) using TCAM cell layout and Cacti parameters for a particular CMOS technology. The value of $C_{\rm metal-wire-per-cell-h}$ depends on the TCAM cell width and metal wire capacitance $(C_{\rm metal})$. For 0.18-μm CMOS technology and a TCAM cell design of height 4.05 μm and width 4.33 μm, we find the values of $C_{\rm metal-wire-per-cell-h}$, $C_{\rm drain-compareT}$, $C_{\rm nandinv-gate}$, and $C_{\rm precharge}$ to be 1.39, 0.9, 2.4, and 8.9 fF, respectively. The last step is that we need to account for the fact that every cycle, *many* match lines are discharging. This is easy enough to handle as we can just scale (2) by a factor equal to the number of rows. Note that in the case of worst case power estimation, it will prove quite accurate as on each cycle every line is toggling from "0" to "1" back to "0."

While we make some simplifications in our model, in Section V, we extract the height and width from the layout

$$C_{matchline}\ (\ in\ \ fF\ ) = (C_{metal-wire-per-cell-h} + 2 * C_{drain-compareT})\ *\ l\ \ + C_{nandinv-gate} + C_{precharge}$$

where, $C_{metal-wire-per-cell-h}$ is the metal wire capacitance per TCAM cell in horizontal direction, which is the product of TCAM cell width and C_{metal}
$C_{drain-compareT}$ is the drain capacitance of the NMOS compare transistor
$C_{nandinv-gate}$ is the gate capacitance of the nand gate and inverter
$C_{precharge}$ is the capacitance of the precharge circuit

$$C_{searchline}\ (\ in\ \ fF\ ) = (C_{metal-wire-per-cell-v} + C_{gate-compareT})\ *\ r\ \ + C_{driver}$$

where, $C_{metal-wire-per-cell-v}$ is the metal wire capacitance per TCAM cell in vertical direction, which is the product of TCAM cell height and C_{metal}
$C_{gate-compareT}$ is the gate capacitance of the NMOS compare transistor
C_{driver} is the capacitance of the driver circuit

Fig. 3. Matchline and searchline capacitance equations by taking into account the capacitance of metal wires and its loading effect.

of several published TCAM cells and compare our power estimates with their measured results.

2) Select Line Power: While the largest amount of power comes from the match lines (as we will show in Section V), the match lines are not the only large capacitances that need to be charged every cycle. With each access the select lines must be charged according to the search query (described in Section III). As the TCAM scales, the size of the select lines grow, and the amount of energy to drive these lines can quickly add up. The select lines are different from the match lines in a couple of respects that prevent them from dominating. As can be seen in Figs. 1 and 2, there is only one transistor per cell that interfaces with this line, and it is connected to the gate. We need to take into account the gate capacitance of one of the nMOS compare transistor while calculating the equivalent capacitance for the search line.

For r number of rows, we can calculate the search line capacitance using (3) as shown in Fig. 3. For (3), we calculate the values of capacitances ($C_{\mathrm{metal-wire-per-cell-v}}$, $C_{\mathrm{gate-compareT}}$, C_{driver}) using TCAM cell layout and Cacti parameters for a particular CMOS technology. The value of $C_{\mathrm{metal-wire-per-cell-v}}$ depends on the TCAM cell height, and metal wire capacitance (C_{metal}). For 0.18-μm CMOS technology and a TCAM cell design of height 4.05 μm and width 4.33 μm, we find the values of $C_{\mathrm{metal-wire-per-cell-v}}$, $C_{\mathrm{gate-compareT}}$, and C_{driver} to be 1.31, 0.103, and 18.31 fF, respectively. One important thing to notice is that *not every line switches every cycle*. To search for a one or a zero in the TCAM cells, only one of the select lines is driven high. To do a "don't care" bit, neither line is driven. The case where $Select_0$ and $Select_1$ are both driven high should never happen. This takes a factor of 2 off of worst case activity factor right away (from 100% down to 50%). While we might be tempted to consider the possibility that sometimes we may search for the same bit at the same position in two contiguous cycles, in the TCAM architecture described in Section III, this cannot happen. We must bring the select lines back to zero before precharging the match lines to prevent a direct path from V_{dd} to ground (from the precharge circuit, through the match-line, through a matching select bit, and to ground). It is conceivable that any future design of TCAM, that does not require the select lines to return to zero, can take advantage of the fact that some bits are less likely to switch than others, but we have estimated from routing table traces that this would likely only reduce the activity factor from 50% to 46%.

3) Priority Encoder Power: The priority encoder also consumes some power and is dependent on the number of rows. The dynamic power consumption in the priority encoder is independent of the number of bits. We take the result from [25] and [26] to estimate the energy consumption of a $N \times 1$ priority encoder, where N is the number of rows. We use a 256-bit priority encoder from [25] and apply 2-level lookahead to design a higher-bit priority encoder. In [25] we find that the energy consumption of a 256-bit priority encoder is 1.5 pJ in 0.18-μm technology. We calculate the number of the 256-bit priority encoders and the primitive gates required for a higher-bit priority encoder. The power consumption of the primitive gates is negligible compared to a 256-bit priority encoder. We calculate the total power consumption of a large priority encoder by multiplying the number of the 256-bit priority encoders required and the energy consumption of one 256-bit priority encoder [25].

B. Read/Write Power

The read/write operation in a TCAM is very similar to SRAM with very minor differences. Instead of reading/writing one cell in SRAM, two SRAM cells are read/written for one TCAM cell. Like SRAM, read/write operation is controlled by the *wordline*. The *bitlines* are used to actually read/modify the stored data in the TCAM cells. The length of the *wordline* and *bitline* wires are the main components, which affects delay and power design constraints. The sense amplifier is used to sense the bit stored in the cell on a read operation. Similarly, the *bitline* driver is used to force the data in the cell on a write operation. We use the SRAM power model inherent to Cacti [22] to model the read/write power, but we use the TCAM cell layout and other parameters so that the length of *wordlines* and *bitlines* are calculated accurately. While we do not include an exploration of read/write power in this paper, this functionality is included as a part of our modeling tool.

C. Leakage Power

Since many past techniques seek to optimize power by enabling only a subset of the full TCAM [1], [2], [19], as we move to deep sub-micron technology this may lead to an increasing percentage of total system power being consumed by leakage from inactive TCAM partitions. We consider sub-threshold leakage power as it is the dominant source of leakage in TCAM structures [27].

We leverage models from existing work [13], [28], [29]. HotLeakage [29] provides a subthreshold leakage model based

on BSIM3. Cacti 4.0 [28] and eCacti [13] then use HotLeakage to provide the static power for memory and cache structures. The leakage equation in HotLeakage tool takes into account the exponential relationship between temperature and leakage, threshold voltage, device width (W) and many other low-level device parameters.

For our TCAM model, all of the parameters are assumed to scale in the same manner as predicted for SRAM. The only parameter that is directly effected by the fact that we are modeling TCAM instead of an SRAM cell, is the device width W. W for the different transistors in the TCAM is derived from a combination of the physical layout and the wire loading calculated from the architectural configuration.

Building on these SRAM models, we estimate the leakage power for major components inside of a TCAM including: 1) TCAM storage transistors; 2) TCAM access transistors; 3) decoder circuit; 4) matchline precharge circuits; and 5) wordline driver circuits. We find that the remaining circuits contribute very little to the overall leakage. To model the leakage power of each component, we consider the leakage power of participating nMOS and pMOS transistors by first calculating the width of the transistors and then using the subthreshold leakage current equation in HotLeakage [29]. This estimate can then by multiplied with the number of participating nMOS and pMOS transistors for a particular component. For example, if the bitlines are kept low, in one TCAM cell there are two leaking access transistors (both nMOS) and four leaking transistors in the memory cell (two nMOS and two pMOS). Then by calculating the equivalent transistor width of each of these leaking devices, we can estimate the overall leakage power. This leakage analysis has also been incorporated in our modeling tool.

D. Delay Modeling

The maximum operating frequency for a TCAM is mainly dependent upon the time of accessing precharge circuit, precharging matchlines, driving searchlines, and priority encoding. Generally, the access time for any circuit is decided by the time constant of the circuit which is a product of equivalent resistance $(R_{\rm eq})$ and equivalent capacitance $(C_{\rm eq})$. We have already calculated the values of equivalent capacitance of the major components in earlier sections and now we calculate the equivalent resistance of each of these components. We use Horowitz's approximation approach (described in [12]) to calculate the access time of precharge circuit, matchline, and searchline, which is provided in (4)

$$\begin{aligned} delay_{\rm rise}(t_f, t_{\rm rise}, v_{\rm th}) &= t_f \sqrt{(log[v_{\rm th}])^2 + 2t_{\rm rise} b(1-v_{\rm th})/t_f} \\ delay_{\rm fall}(t_f, t_{\rm fall}, v_{\rm th}) &= t_f \sqrt{(\log[1-v_{\rm th}])^2 + 2t_{\rm fall} b v_{\rm th}/t_f} \end{aligned} \tag{4}$$

where

$v_{\rm th}$	switching voltage;
$t_{\rm rise}$, $t_{\rm fall}$	input rise time and output fall time, respectively;
t_f	input/output time constant;
b	fraction of swing in which input affects the output.

We provide the support of sub-banking and row divisions $(N_{\rm rd})$ in our model in order to get the optimal delay for a particular TCAM size. The overhead of sub-banking the large TCAM including routing, and buffer sizing is taken into account. Using a multilevel design, we find the level of 256-bit priority required in a chain of priority encoders for large number of rows. We use the results from [25] where a 256-bit priority encoder requires 0.367 ns in 0.18-μm technology. Usually, TCAM search operation is pipelined with priority encoder in a different pipeline stage. Hence, the *critical-path* delay for larger TCAMs is mostly dictated by the access time of matchline and searchline.

While the delay model can provide the maximum operating frequency of the TCAM structure, the TCAM circuit can be run at a lower speed to minimize the power consumption. Hence, we can have different power consumption for different operating frequency. Later, we present the delay analysis only to show the maximum frequency that can be achieved and we limit our discussion to energy consumption per access.

V. Evaluation

Building on the work done in Cacti [12], and the capacitive models developed in Section IV, we have created power estimator for TCAM that is parameterizable and simple to use for either relative or absolute comparisons. In this section, we evaluate our model against several physical implementations, and describe the scaling of various components in TCAM. For convenience, our model has been coded into a simple tool which is made publicly available at http://www.cs.ucsb.edu/~arch/memmodel.

While, thus far, we have described the power modeling of TCAM at a theoretical level, our goal is to enable fair comparisons not only between different TCAM designs, but also among TCAM, Memory, and even logic. To calculate such parameters as the capacitance of a wire per unit length in real physical units $(\rm fF/\mu m)$ we need to use the characteristics of existing process technology and VLSI implementations. We collected these parameters from existing tools, VLSI layouts, and published results. In particular we base our parameters on the following:

- *Cacti tool* [12]—Cacti provides a number of low-level circuit parameters for 0.80-μm CMOS technology. We scale the parameters appropriately for 0.18-μm CMOS technology. Some of these parameters include the capacitance of metal wires, gate capacitance per unit area etc. To verify that the scaled values are not significantly different than current technology we checked them against a published literature. For example, the wire capacitance $(C_{\rm metal})$ from [30] is 0.25-fF/μm, which is within 10% of our estimate. For future technologies, we incorporate the wire features from ITRS roadmap [24] to estimate this parameter.
- *TCAM layout*—While Cacti is a useful start, and will ensure fair comparisons between Cacti results and our model, it is limited in its usefulness because it assumes a 6-transistor static SRAM cell, not a much larger TCAM cell. We collect the remaining required values from a published static TCAM cell layout [14]. Parameters extracted from this design include TCAM cell width, cell height, and the width of the transistors used in the comparison operation.

As network algorithm and architecture designers prefer to concentrate on the high-level architectural parameters to

optimize their design, our model takes all the high-level architectural parameters such as *number-of-rows*, *column-bits*, *number-of-banks*, and *CMOS technology* as the input parameters and provides the worst case dynamic energy consumption. While these parameters are most directly applicable to an architect, TCAM cell design is not a solved problem and it is still progressing. Because of this there can be variations in the height and width of a cell and we leave these as optional parameters (the default height and width are extracted from [14]). For example, a Dynamic TCAM [16], [17] increases the effective number of bits per area because it uses only 1-transistor dynamic cell to store a bit. When our model is given the height and width of dynamic TCAM cell, we can estimate the power accurately even though the bit storage is completely different. We now demonstrate that our power model matches closely with published implementations and describe the effect of architectural parameters on the dynamic power of TCAM in a search operation.

A. Validation of Our Model

We verify our model with as many physical implementations that we could collect from industry datasheets and circuits conferences.

Arsovski and Wistort [31] present a 64×240 bit TCAM design in 65-nm technology and the power consumption for this design is found to be 10 mW at 450 MHz. Using our model, we find that the same TCAM size in 65-nm technology node consumes 8.2 mW at the same frequency, which is within 20% of the published results.

SibreCore Technologies, a producer of TCAMs, states in a white paper [32] that SibreCore's SCT2000 consumes less than 1.7 W/Mb without any active power management and at a frequency of 66 MHz. We used our model to estimate the power consumption of a 1-Mb TCAM in 0.18-μm CMOS technology with 2.5-V supply voltage which we believe is close to their design technology. Our model predicted a power of 1.85 W/Mb, a percent difference of less than 8% from the published results. Given that we do not have precise technology or layout parameters, we believe this to be a close fit. Analog Bits, a TCAM vendor, markets a 512×144 TCAM, which runs at 800 MHz and consumes 0.5-mA/MHz current [33]. This TCAM is available for TSMC CL013LV/LVOD process. We use the same process features to find the power consumption for this TCAM configuration using our model and we find that it consumes 0.53-mA/MHz current, which is with in 6% of the published results. Using our delay model, we find that using four subbanks this TCAM configuration can run at maximum of about 840 MHz.

Noda *et al.* [17] finds that the power dissipation for a conventional 4.5 Mb static TCAM without any power management features is approximately 7.0-W assuming a supply voltage of 1.5 V and operating frequency of 143 MHz. We feed these high-level architectural parameters to our model and we find that the power dissipation is 6.4 W, which is again close. To verify the feasibility of operating frequency, we find the access time for this TCAM configuration, which gives a maximum operating frequency of 188 MHz using 16 subbanks. Noda *et al.* [17] also provides the power dissipation of a dynamic TCAM (2 W) and TCAM cell features. We use these dynamic TCAM cell features to measure the power dissipation of the dynamic TCAM and it is found to be 2.712 W. We also compare the relative contributions of each hardware component ($matchline$, $searchline$, etc.) from [17] and we find that the results are accurate with in 10%.

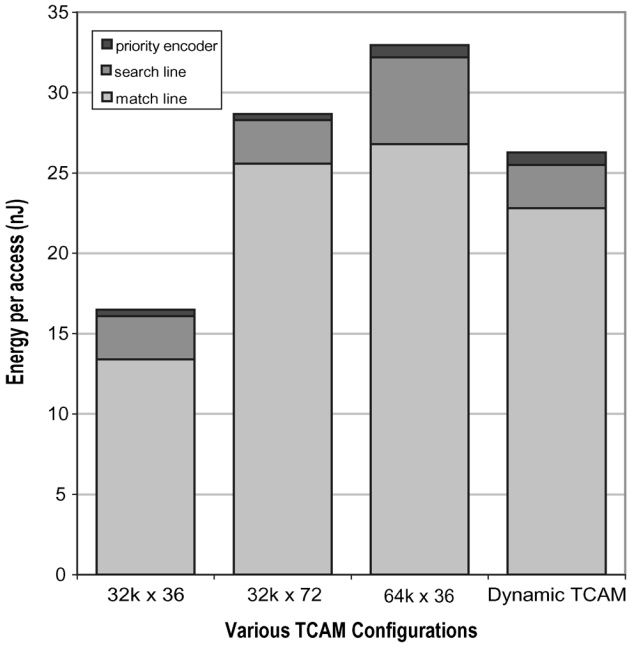

Fig. 4. Breakdown of the energy consumption per access from the priority encoder, search line, and match line for a variety of TCAM configurations in 0.18-μm CMOS technology. The Dynamic TCAM is 64K × 36, with 1-T cells and its cell layout is used in power estimation.

Mohan and Sachdev [34] present techniques to reduce the static power in ternary CAMs. From this work, we find that the TCAM cell leakage current in 70-nm technology at room temperature is found to be about 45 nA, whereas our leakage model estimates 64-nA leakage current for one TCAM cell. This difference may be due to the static power reduction schemes employed in [34]. Since we could find only one published results on TCAM leakage power, it further necessitates the need of such a modeling tool so that network designers can evaluate various power management schemes using this model.

B. Energy Breakdown

Fig. 4 shows the breakdown of the energy consumed for a single access of the TCAM for a variety of different configurations. All of the configurations are in 0.18-μm at 1.8 V. The first configuration has 32 K entries, each of length 36, and requires a total of 16.7 nJ for search access. The majority of the power, as expected, is being consumed by the match line, but there is still a noticeable impact from both the search line and priority encoder. As a point of comparison, an SRAM of similar size would require approximately 1.9 nJ for a simple read operation. While an SRAM read will happen much faster, if more than 8.7 memory accesses are required on average to perform the same job, a TCAM could actually be a lower power system.

The second configuration shows the effect of doubling the size of an entry (from 36 to 72) and the third configuration shows the effect of doubling the *number* of entries. Surprisingly, adding more entries into a single bank of TCAM costs more per bit in terms of power than extending the size of an entry. We can

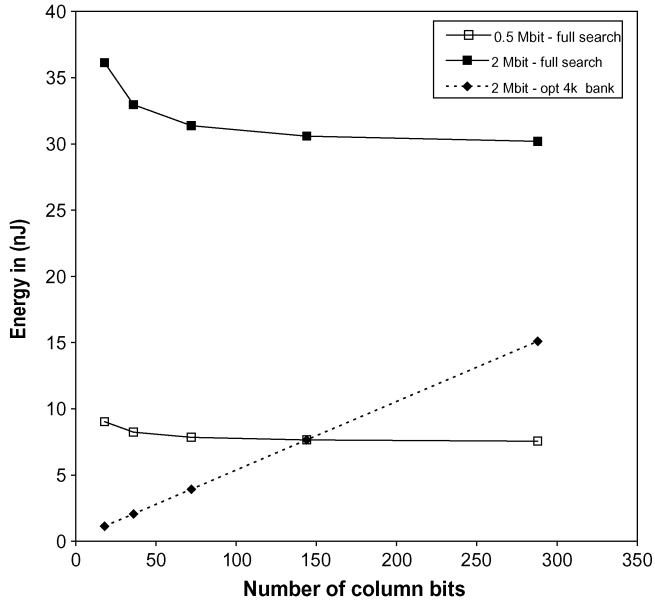

Fig. 5. Scaling of energy per access with the length of the TCAM entry in 0.18-μm technology node. For a monolithic design, the energy per access actually drops as the length is increased slightly, but this comes at the expense of long latencies. An optimally banked design scales almost linearly with the size of an entry.

see that between these two configurations, the match line power is roughly equivalent, but because of the longer search lines the search line power grows significantly. Doubling the number of entries also requires more energy in the priority encoder to select among twice as many possibilities. The final configuration is the Dynamic TCAM which is a 64 K $\times$ 36 configuration and is comparable to the third configuration. The reduced size of the device results in shorter match and select lines, although the sum of the capacitive loading from the transistors is unchanged due to the fact that an equal number of comparisons still need to be made. Instead the transistors are just packed onto shorter lines. The impact of the reduced line size decreases both the select and match line power but leaves the priority encoder unchanged.

C. Effect of Parameters on Energy

The results from Section V-B introduced an interesting phenomenon. For a fixed size TCAM (in terms of number of cells) a larger entry size (column bits) can sometime lead to *less* energy per access than a tall and narrow TCAM bank. To explore this idea further, we present Fig. 5 which shows the variation in energy per access for varying number of columns.

Two different sizes are shown, both for 0.18-μm technology with a fixed total size. On the x-axis, we show the effect of scaling the size of the entries, but because the total size in bits is fixed, the number of entries will be reduced as the entry size increases. While the energy per access is quite flat across a wide range on entry sizes, it climbs sharply below 32. Many vendors choose to ship a device that has a long maximum entry size (for example 72 or 144 entries), that is then configurable to smaller entries as needed.

The two configurations discussed assume that there is no subbanking and no dynamic power management to reduce the energy per access on a large TCAM. If instead the TCAM was broken into banks, each with 4K entries, this could potentially reduce the power significantly. The problem is knowing *exactly* which bank your data resides in *without* searching through them. The dashed line in Fig. 5 shows exactly this. The energy per access scales linearly with the length of an entry because there will be less and less possible banks across which we can divide the data. These banked energy numbers are not meant to be directly achievable, but rather serve as a guide to algorithm designers seeking to trade off smart ways of controlling TCAM banks with the underlying dynamics of the hardware.

In Fig. 6, we show the effect of technology and voltage scaling on the total power for TCAM search. As the feature size drops, and the voltages are scaled, the power for searching through a TCAM has dropped significantly. We also see the similar decreasing trend for matchline, searchline and priority encoder as we decrease the CMOS feature size. A 100-K entry TCAM in 0.4-μm technology requires more than a factor of 10 times more joules/access than a comparable design in 90 nm.

D. Leakage Power

Based on the analysis described in Section IV-C, we calculate the leakage power for a 32768 $\times$ 36 TCAM configurations for different technology nodes at 100 degree Celsius (373 K). Fig. 7 shows the leakage power of 32 768 $\times$ 36 TCAM for 130-, 100-, and 70-nm technology nodes. We divide the leakage power into five main contributing components as shown in Fig. 7. As obvious, the total leakage power is very low in 130-nm technology and it increases sharply as we move into deep submicrometer technology (70 nm). To quantify it further, as we move from 130 to 100 nm, the total leakage power increases by almost *seven* times and when we move from 100 to 70 nm, the total leakage power increases by about *four* times. We find that major leakage power comes from cell access and cell storage transistors as these transistors occupy most of the chip area. For example, in 70-nm technology node, the cell access and storage transistors contribute more than 75% of total leakage power. Comparing this with the dynamic power of the same TCAM configuration operating at 400 MHz, we find that the total leakage power is only about two times less than the dynamic power. Even for TCAM, leakage will become a major fraction of the total power consumption in deep submicrometer technologies and at higher temperatures.

E. Effect of Parameters on Delay

In order to get the maximum power consumption, we also need to calculate the maximum clock frequency. In this section, we present the delay analysis of TCAM to find the maximum attainable clock frequency. Consequently we can get the worst-case power using the energy results described in the previous sections.

Fig. 8 shows the effect of TCAM size on the access time in 90-nm technology. We vary the size of TCAM from 8 kbits to 4 Mbits. For all these TCAM configurations, we select the optimal number of banks and number of row divisions to minimize the access time while also keeping the power consumption low. We find that for smaller sizes of TCAM, the increase in delay is very small, but for larger TCAM the delay increases sharply. For example, when we double the TCAM size from 32 to 64 kbits, the increase in delay is about 13%. But if we increase it from 2 Mbits to 4 Mbits, the increase in delay is found to be about 21%. This is mainly because we cannot continue to do further

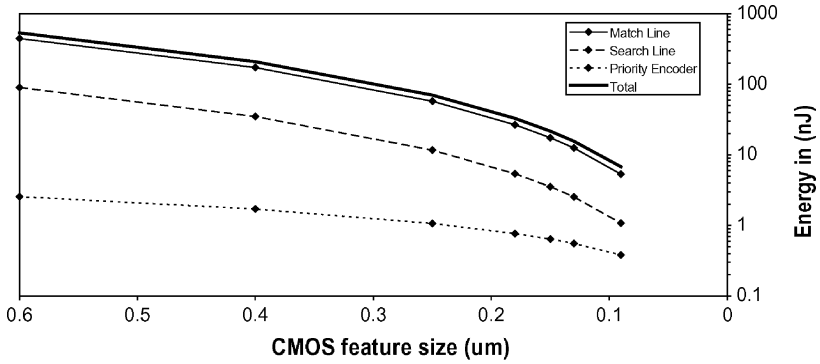

Fig. 6. Scaling of energy consumption per access for a 100 K × 36 TCAM as feature size and voltage are scaled back with time. The y-axis is in log scale.

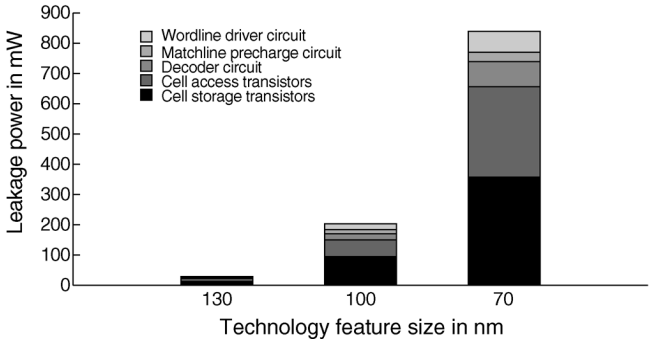

Fig. 7. Leakage power for a 32 768 × 36 TCAM configuration for different technology nodes at 373-K temperature. The leakage power has been classified into five main contributing components as shown.

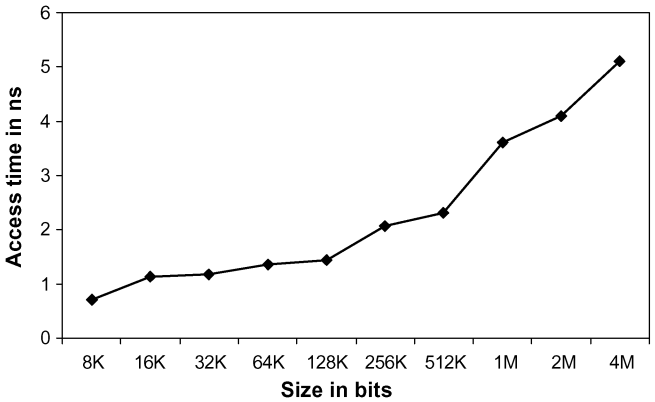

Fig. 8. Effect of TCAM size on the access time in 90-nm technology. TCAM size is varied from 4 kbits to 8 Mbits. While doubling TCAM sizes increases the delay slightly for smaller sizes (less than 128 kbits), there is a larger increase in the access time for bigger TCAMs.

subbanking and row divisions for larger TCAMs as it directly increases the overall power and area overhead.

F. Uses for the TCAM Model

While our model is directly useful to network device developers, it can also be easily extended for various power optimization techniques. The extra work required to do this is minimized because our model is decomposed into the various components that are most likely to be targets for optimization. The impact of optimizations can be easily estimated by changing the appropriate parameters in the model.

1) Extending the Model for Circuit Level Optimizations: For example, the matchline power consumption can be reduced by using current sensing to enable low voltage swings [14]. We can multiply our already calculated equivalent capacitance with $(\delta V) * V_{\rm dd}$, where δV is the voltage swing. We also need to find the equivalent capacitance of sense amplifier circuit to find the total power consumption. Similarly, with little effort, our technique can also be extended to model other power optimization methods such as low power dual matchline [35] and pipelined matchlines with hierarchical searchlines [36]. In the case of dual matchline [35], the matchline is divided into two segments to reduce the active capacitance and the second matchline segment is only enabled if there is a match in the first segment. This can be easily incorporated into our model by updating the matchline equations with conditional matching and including the extra sense amplifier circuits.

In pipelined matchlines with hierarchical searchlines [36], the matchlines are divided into five pipelined segments while searchlines are divided into global and local searchlines. The only local searchlines that are enabled are those that may potentially match, based on information from previous pipeline segments. This can be modeled in our tool by modifying the sizes of the matchline and searchline and considering the extra sense amp overhead.

2) Using the Model for Algorithm Level Analysis: Based on the use of TCAM in different networking applications, we can find the activity factor of different components based on application traces. As we show in the modeling section (see Section IV), the activity factor of both the matchline and priority encoder is close to 100%, while for searchlines it is about 46%–50%. Depending on the target deployment and real traffic encountered there, the designer can find the exact activity factor for each component. For example, for IP forwarding application in network routers, one can use IP address traces and the routing table to find the exact switching activity for each of the components. While there will be only a very marginal decrease in power compared to the worst case power for this application, for other algorithms the savings may be more significant.

G. Uses for Cross-Level Tradeoffs

Doing an absolute study is very critical when there are many implementations possible for an algorithm using different types of memory. TCAM can provide O(1) search and high throughput, but the designers may opt for a SRAM-based implementation to meet a more strict power budget. But there

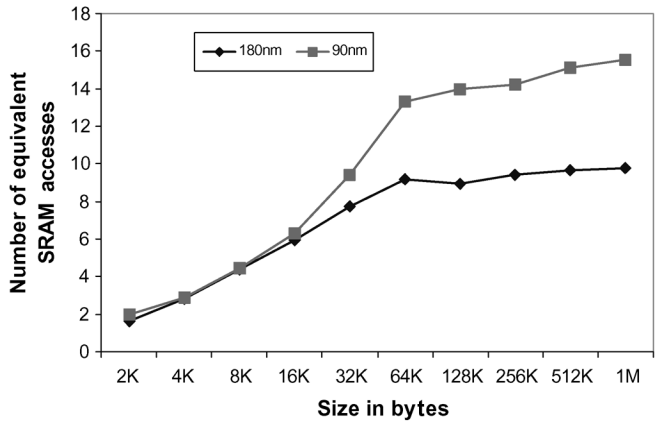

Fig. 9. Number of equivalent SRAM accesses compared to one TCAM access with the same energy consumption in 180- and 90-nm technologies. More than 15 SRAM accesses in 90-nm technology for a network algorithm implementation can make the TCAM-based solution more power effective.

are techniques to narrow down the energy gap between TCAM and SRAM which prune the search space in TCAM. On top of that SRAM usually runs at higher clock frequency and a SRAM based implementation may require multiple accesses compared to one access in TCAM. Hence, we need a comparative study of SRAM and TCAM that will be extremely helpful for the network designers to explore various networking algorithms and power management schemes.

1) SRAM and TCAM Comparison: To do a comparative study between SRAM and TCAM energy consumption, we also develop a SRAM power model. We modify Cacti [22] to get the power consumption of SRAM data array instead of total cache power. We vary the size of SRAM/TCAM from 2 kbytes to 1 Mbytes and find the energy consumption. In this comparison we assume that SRAM and TCAM implementation of an algorithm may require almost the same amount of memory. While this assumption is just for this case study, but possible variations of different sizes of SRAM and TCAM can certainly be studied using our models. Both the SRAM and TCAM assume the same core 6T memory cell to allow for a direct comparison.

Fig. 9 shows the SRAM and TCAM energy comparison for different sizes in 180- and 90-nm technologies. The y-axis shows the number of equivalent SRAM access required to have the same energy consumption of one TCAM access. We find that for smaller size we need less than five equivalent SRAM accesses for one TCAM access for both the technology nodes. But as we increase the size, the number of equivalent SRAM accesses increases up to about 10 in 180-nm and about 15 in 90-nm technology. This shows that SRAM are being more power-optimized in current technologies by optimizing various circuit components including cell structure and sense amplifiers. For 128 kbytes and greater in 90-nm technology, if an algorithm requires 15 SRAM accesses compared to one TCAM access, then TCAM solution can be more energy effective. For example, for a simple trie implementation of a packet classification application, the lookup operation may require up to 20 SRAM accesses. The same lookup can be done using just one TCAM access. We believe that enabling such direct comparisons between SRAM and TCAM, these models will open the door for researchers to explore and combine them in novel and interesting ways.

VI. Conclusion

In high-speed networking applications, TCAM has been used as one of the principal components due to its ability to perform fully associative ternary search. This ability can be exploited to perform a wide range of operations, and new applications are still being discovered and implemented. To provide a fair comparison against past techniques when power is concerned, there is a need for an accurate TCAM power model that can be directly compared against comparable SRAM, cache, and logic models.

In this paper, we have shown that such a model can be built through the calculation of match and select line capacitances by considering both the wire length and loading of these lines. Our model can factor in operating frequency, number of entries, length of entries, banks, and even circuit level parameters such as cell height and width. We describe how TCAMs scale with these parameters and validate our model against several physical designs. Our model can also be easily extended to find dynamic power consumption based on some specific networking traces or to incorporate some circuit-level optimizations.

We show that the energy to search a TCAM is not significantly more than the energy consumed by several SRAM accesses to a memory of comparable size. This sort of comparison will provide a foundation on which to do hybrid SRAM/TCAM algorithms research. We believe our model will enable researchers to find and exploit tradeoffs at the algorithm and architecture level, and will enable realistic energy estimations to be made across a wide range of TCAM-based applications and designs.

References

[1] G. J. Narlikar, A. Basu, and F. Zane, "CoolCAMs: Power-efficient TCAMs for forwarding engines," in *Proc. IEEE INFOCOM: Conf. Comput. Commun.*, 2003.

[2] V. C. Ravikumar, R. Mahapatra, and L. Bhuyan, "EaseCAM: An energy and storage efficient TCAM-based router architecture for IP lookup," *IEEE Trans. Comput.*, vol. 54, no. 5, pp. 521–533, May 2005.

[3] H. Liu, "Efficient mapping of range classifier into ternary-CAM," presented at the 10th Symp. High Performance Interconnects HOT Interconnects (HotI'02), Stanford, CA, Aug. 2002.

[4] K. Zheng, C. Hu, H. Lu, and B. Liu, "An ultra high throughput and power efficient tcam-based ip lookup engine," in *Proc. INFOCOM: Conf. Comput. Commun.*, 2004, pp. 1984–1994.

[5] E. Spitznagel, D. Taylor, and J. Turner, "Packet classification using extended TCAMs," in *Proc. 11th IEEE Int. Conf. Network Protocols (ICNP)*, 2003, p. 120.

[6] F. Yu, R. H. Katz, and T. V. Lakshman, "Gigabit rate packet pattern-matching using TCAM," presented at the 11th IEEE International Conference on Network Protocols (ICNP), Berlin, Germany, 2004.

[7] R. Panigrahy and S. Sharma, "Sorting and searching using ternary CAMs," *IEEE Micro*, vol. 23, no. 1, pp. 44–53, Jan./Feb. 2003.

[8] N. Bandi, S. Schneider, D. Agrawal, and A. E. Abbadi, "Hardware acceleration of database operations using content addressable memories," presented at the 1st Int. Workshop on Data Management on New Hardware (DaMoN), Baltimore, MD, 2005.

[9] Cisco Systems, , , "600-Watt redundant AC power system," 1998.

[10] Cypress Semiconductors, Laguna Hills, CA, "Ayama(tm) 10000a Network Search Engine," 2004.

[11] H. Wang, L. Peh, and S. Malik, *A Power Model for Routers: Modeling Alpha 21364 and Infiniband Routers*. Stanford, CA: Morgan Kaufman, Aug. 2002.

[12] S. Wilton and N. Jouppi, "An enhanced access and cycle time model for on-chip caches," DEC Western Res. Lab, Tech. Rep. 93/5, 1994.
[13] M. Mamidipaka and N. Dutt, "eCACTI: An enhanced power model for on-chip caches," Tech. Rep. CECS TR-04-28, Sep. 2004.
[14] I. Arsovski, T. Chandler, and A. Sheikholeslami, "A Ternary Content-Addressable Memory (TCAM) based on 4T static storage and including a current-race sensing scheme," *IEEE J. Solid-State Circuits*, vol. 38, no. 1, pp. 155–158, Jan. 2003.
[15] K. Pagiamtziz and A. Sheikholeslami, "A low-power content-addressable memory (CAM) using pipelined heirarchical search engine," *IEEE J. Solid-State Circuits*, vol. 39, no. 9, pp. 1512–1519, Sep. 2004.
[16] V. Lines, A. Ahmed, P. Ma, S. Ma, R. McKenzie, H. Kim, and C. Mar, "66 MHz 2.3 M ternary dynamic content addressable memory," presented at the 8th IEEE Int. Workshop on Memory Technology, Design, and Testing (MTDT 2000), San Jose, CA, 2000.
[17] H. Noda, "A cost-efficient high-performance dynamic TCAM with pipelined hierarchical searching and shift redundancy architecture," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 245–253, Jan. 2005.
[18] S. Sharma and R. Panigrahy, "Reducing TCAM power consumption and increasing throughput," presented at the 10th Symp. High Perf. Interconnects HOT Interconnects (HotI'02), Stanford, CA, Aug. 2002.
[19] X. Li, Z. Liu, W. Li, and B. Liu, "SCP-TCAM: A power-efficient search engine for fast IP lookup," presented at the ISBN Proc., 2004.
[20] H. Wang, X. Zhu, L. Peh, and S. Malik, "Orion: A power-performance simulator for interconnection networks," in *Proc. Int. Symp. Microarch.*, Nov. 2002, pp. 294–305.
[21] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in *ISCA '00: Proc. 27th Annu. Int. Symp. Comput. Arch.*, New York, 2000, pp. 83–94.
[22] P. Shivakumar and N. P. Jouppi, " 3.0: An integrated cache timing, power and area model," Tech. Rep. Western Research Lab. (WRL), 2001–2002.
[23] I. Hsiao, D. Wang, and C. Jen, "Power modeling and low-power design of content addressable memories," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2001, vol. 4, pp. 926–929.
[24] International Technology Roadmap for Semiconductors (ITRS), "Interconnect Report," 2006.
[25] W. W. Fung and M. Sachdev, "High performance priority encoder for content addressable memories," in *Proc. Micronet R&D Annu. Workshop*, 2004.
[26] J. Wang and C. Huang, "High-speed and low-power CMOS priority encoders," *IEEE J. Solid-State Circuits*, vol. 35, no. 10, pp. 1511–1514, Oct. 2000.
[27] N. Mohan, "Low-power high-performance ternary content addressable memory circuits," Ph.D. dissertation, Dept. ECE, Univ. Waterloo, Waterloo, ON, Canada, 2006.
[28] D. Tarjan, S. Thoziyor, and N. P. Jouppi, Cacti 4.0 HPL Tech. Rep. HPL-2006-86, Jun. 2006.
[29] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron, and M. Stan, "Hotleakage: A temperature-aware model of subthreshold and gate leakage for architects," Dept. Comput. Sci., Univ. Virginia, Tech. Rep. Cs-2003-05, Mar. 2003.
[30] M. Horowitz, R. Ho, and K. Mai, "The future of wires," presented at the SRC Conf., Stanford, CA, May 1999.
[31] I. Arsovski and R. Wistort, "Self-referenced sense amplifier for across-chip-variation immune sensing in high-performance content-addressable memories," in *Proc. IEEE Custom Integr. Circuits Conf.*, 2006, pp. 453–456.
[32] SibreCore Technologies, "Classification and forwarding co-processors come of age—The myths and realities of high performance ternary CAMs (TCAMs)," Mar. 2002.
[33] Analog Bits, "High speed Ternary CAM datasheet," 2004.
[34] N. Mohan and M. Sachdev, "A static power reduction technique for ternary content addressable memories," in *Proc. IEEE CCECE*, May 2004, pp. 711–714.
[35] N. Mohan and M. Sachdev, "Low power dual matchline ternary content addressable memory," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2004, pp. 633–636.
[36] K. Pagiamtzis and A. Sheikholeslami, "Pipelined match-lines and hierarchical search-lines for low-power content-addressable memories," in *Proc. IEEE Custom Integr. Circuits Conf.*, 2003, pp. 383–386.

Banit Agrawal (S'03) received the B.Tech. degree in instrumentation engineering from the Indian Institute of Technology, Kharagpur, India, in 2001 and the M.S. degree in computer science from the University of California, Riverside, in 2004, and is currently pursuing the Ph.D. degree in computer science from University of California, Santa Barbara.

His research interests include memory design and modeling for network processors, 3-D ICs, security/analysis processors, nanoscale architectures, and power-aware architectures.

Mr. Agrawal was a recipient of a Best Paper Award in the International Symposium on Code Generation and Optimization (CGO) 2006, a Best Paper Nomination in the 39th Annual IEEE/ACM International Symposium on Microarchitecture, 2006, and an IEEE Micro Top Pick Award in Computer Architecture Conferences in 2006.

Timothy Sherwood (S'98–M'03) received the B.S. degree from University of California at Davis, in 1998, and the M.S. and Ph.D. degrees from University of California at San Diego, San Diego, in 2003, where he worked with Prof. B. Calder.

He joined the University of California at Santa Barbara (UCSB), Santa Barbara, in 2003, where he is currently an Assistant Professor. His prior work on program phase analysis methods (a technique for reasoning about and predicting the behavior of programs over time has been cited over 350 times and is now used by Intel, Hewlett Packard (HP), and other industry partners to guide the design of their largest microprocessors. His research interests focus on the area of computer architecture, specifically in the development of novel high throughput methods by which systems can be monitored and analyzed.

Prof. Sherwood was the recipient of a National Science Foundation (NSF) CAREER Award in 2005, and, for the three consecutive years since joining UCSB, he has received the IEEE Micro Top Pick Award for novel research contributions of significance to the computing industry.