ILP formulation **July 2015**

RMT — ILP compiler for P4

Lisa Yan (yanlisa@stanford.edu)

Problem overview

The compiler problem is to optimally fit a P4 program into the RMT switch. A later document will include the specifications for a FlexPipe switch.

- **Program**: A P4 program is a group of logical tables, whose control flow graph be represented in a Table Dependency Graph. The dependency between tables defines whether or not they can be executed in parallel: if there is no dependency, the tables can be reordered without affecting the outcome. However, match dependencies (where a table matches on a field that an earlier table modifies), for example, force tables to be in separate physical stages. Other types (action, successor, and reverse-match) are detailed in the attached NSDI paper, Table 1. The type of dependency not only affects the table placement but also the overall latency of the pipeline execution, which is one of our optimization objectives.

- **Switch**: The RMT switch is a series of pipeline stages, each with two types of memory arranged in what we will call *memory blocks*: SRAM (exact memory type) and TCAM (ternary memory type, which supports wildcard, prefix, and exact match). Each block can hold a set amount of bits; for example, each SRAM block can hold at most 1000 entries of at most 80 bits wide (more information is in Figure 5c in the attached NSDI paper). These blocks can be linked together to match upon larger words at once; a 120 bit word can be matched in two SRAM blocks (a max width of 160 bits, 1000 entries). Furthermore, logical table words can be *packed* together to match multiple words at a time in a memory block; a 40 bit word with a *packing factor* of 2 produces a longer word of 80 bits, so an SRAM block can actually hold 2000 of these 40 bit words with this packing factor; we call this $80b \times 2000$ unit a *packing unit*, which occupies exactly one SRAM block. A packing factor of 5 for a 48-bit word gives a packing unit that occupies three SRAM blocks. More options are shown in Figure 6a attached.

- **Action memory**: Logical tables need scratch space for their actions, which must be allocated in SRAM (not TCAM!) of the same stage. We require that for a given number of entries n in SRAM or TCAM, there must be n action memory entries in SRAM; action entry width is the width of the maximum modified header field in the table. Action entries cannot be packed together; so 2000 40-bit action entries will occupy two SRAM blocks.

- **Crossbars**: There is a limit on the number and size of information that can be matched per stage. Per stage, we limit the number of tables, the total width of the fields matched, and the total width of the fields modified.

- **Objectives**: While there are many options for fitting the set of logical tables from the P4 program into an RMT switch pipeline, we would like to find an optimal one. Objective functions include minimizing the number of stages used, the power usage, and the pipeline latency. While the first objective is simple and the second objective is simply a function of the number of SRAM or TCAM memories used, the third objective is more complex.

- **Latency Objective**: Typically, if two stages have tables that have no dependencies between them (as defined by the table dependency graph), then the stages can execute immediately one after the other, with only a separation of one cycle. So a 32-stage pipeline could potentially take $31 + 12$ cycles, where the 31 comes from the separation cycles, and 12 cycles is the time it takes to execute the last stage (or any stage). However, if two successive stages contain tables with a match dependency, then the second stage must wait until the first has completed (12 cycles) before it starts executing. So a 32-stage pipeline could at worst take 32×12 cycles to complete. More latency types are illustrated in Figure 7; in summary, the tables in each physical stage define the inter-stage latency, affecting the overall latency.

Notation

- Logical tables: $l \in \{1, \cdots, N\}$, where N is the number of tables in the P4 program

- Packing factors (for a logical table): $p \in \{1, \cdots, p_{max}\}$, where $p = p_{max}$ is the maximum number of words that can be matched at a time, and the width of the word is multiplied by p_{max}.

- Table dimension: (f_l, e_l, a_l), differing a bit from the paper. f_l is the width of match entry, e_l is the number of match entries, and a_l is the width of action entries.

- Memory types: $m \in \{1, 2\}$, where $m = 1$ is SRAM, and $m = 2$ is TCAM. Let $m_{max} = 2$.

- Action memory: a, used instead of m to indicate action memory assignment.

- Stages: $s \in \{1, \cdots, M\}$, where there are $M = 32$ stages in the RMT pipeline.

- Block dimension: (b_m, w_m, d_m), where for a stage s and memory m, b_m is the number of blocks, w_m is the width of these blocks, and d_m is the depth (number of entries) each block holds.

- Binary and aggregation variables: $\hat{U}_{l,s,m}$, $\widetilde{U}_{l,s}$, and $\hat{\widetilde{U}}_{l,s}$ for example. $U_{l,s,m}$ is the number of blocks assigned to table l in stage s, memory m, and $\hat{U}_{l,s,m}$ designates a boolean of

whether or not there are any blocks assigned in memory m; $\hat{U}_{l,s,m} = \mathbb{1}\{U_{l,s,m}\}$. $\widetilde{U}_{l,s}$ is the total blocks assigned for table l to any memory in stage s, and $\hat{\widetilde{U}}_{l,s}$ is a boolean of whether any blocks are assigned to any memory; $\hat{\widetilde{U}}_{l,s} = \mathbb{1}\{\widetilde{U}_{l,s}\} = \mathbb{1}\{m \in \{1,2\} : \hat{U}_{l,s,m}\}$.

- Product variables: $P_{l,m}^{U,S}$ indicates the product of U and S variables, both of which have reference subscripts l and m.

Variables

- Memory assignment variables.
  Count: $(m_{max}+1)(3NM) + (2m_{max}+1)(NMp_{max}) = O(m_{max}NMp_{max})$
  - $U_{s,l,m}$ blocks, the number of blocks in stage s dedicated to table l. Match memory only.
  - $W_{s,l,m}$ words, the maximum number of match entries of table l in stage s based on the assignment $U_{s,l,m}$.
  - $T_{s,l,m,p}$ packing units. The conversion of $T_{s,l,m,p}$ to $U_{s,l,m}$ and $W_{s,l,m}$ can be precomputed by a preprocessor, since it is only a function of the packing factor p.
  - Action memory variables: $U_{s,l,a}$ blocks and $W_{s,l,a}$ words, both a function of $T_{s,l,a,1}$; note $p = 1$ always for action memory.
  - Binary block variables: $\hat{T}_{s,l,m,p}, \hat{U}_{s,l,m}, \hat{\widetilde{U}}_{s,l}$, where $\hat{\widetilde{U}}_{s,l}$ is the indicator of whether table l has any memory assigned in stage s, regardless of SRAM or TCAM. Action memory does not count here.
- Dependency variables. Count: $2(NM) + 2(NM)^2 = O(N^2M^2)$
  - $\hat{S}_{s,l}$, indicator if stage s is the first stage that has match memory of l, TCAM or SRAM; "start stage."
  - $\hat{E}_{s,l}$, indicator if stage s is the last stage that has match memory of l, TCAM or SRAM; "end stage".
  - $P_{s,l}^{\hat{S},\hat{\widetilde{U}}}$ Product of $\hat{S}_{s,l}$, whether stage s the start stage of and $\hat{\widetilde{U}}_{s,l}$ is whether there are any table l match memory blocks in stage s.
  - $P_{s,l}^{\hat{E},\hat{\widetilde{U}}}$ Product of $\hat{E}_{s,l}$, whether stage s the end stage of and $\hat{\widetilde{U}}_{s,l}$ is whether there are any table l match memory blocks in stage s.
- Maximum stage variables. Count: $2M + M^2 = O(M^2)$
  - $\hat{\sigma}_s$, indicator if stage s is the last stage ("maximum stage") of the pipeline with table assignment.

  - $\hat{B}_s$, indicator if stage s has any memory blocks assigned across all memories and tables.
  - $P_s^{\hat{\sigma},\hat{B}}$ Product of the maximum stage indicator $\hat{\sigma}_s$ and the memory blocks assigned indicator $\hat{B}_s$.

- Latency variables. Count: $M + 2(NM^2) = O(NM^2)$
  - t_s, start time of stage in terms of latency cycles. For example, the first stage starts at time $t_0 = 1$. All tables l in stage s have the same start time.
  - $P_{s,l}^{\hat{S},t}$, product of start time of stage t_s and the indicator if stage s is table l's start stage, $\hat{S}_{s,l}$. If table l is not in stage s, $P_{s,l}^{\hat{S},t} = 0$.
  - $P_{s,l}^{\hat{E},t}$, product of end time of stage t_s and the indicator if stage s is table l's end stage, $\hat{E}_{s,l}$.

Constraints

Basic constraints

These are constraints that relate the memory assignment variables to each other.

- Word layout constraints: converting $T_{s,l,m,p}$ to $U_{s,l,m}$ and $W_{s,l,m}$ via a preprocessor. The preprocessor returns $f_U(T, s, l, m, p)$ and $f_W(T, s, l, m, p)$, which are fixed functions of the number of blocks and words respectively from choosing packing unit p in stage s, memory type m for table l. Performed for SRAM, TCAM, and action memory. Recall action memory has only one packing factor.
  Count: $2(m_{max}NMp_{max}) + 2NM = O(m_{max}NMp_{max})$

$$U_{s,l,m} == \sum_p T_{s,l,m,p} \cdot f_U(T, s, l, m, p)$$

$$W_{s,l,m} == \sum_p T_{s,l,m,p} \cdot f_W(T, s, l, m, p)$$

$$U_{s,l,a} == T_{s,l,a,1} \cdot f_U(T, s, l, a, 1)$$

$$W_{s,l,a} == T_{s,l,a,1} \cdot f_W(T, s, l, a, 1)$$

- Packing unit constraint: Each logical table l is restricted to one unique packing unit p in stage s. Using binary packing unit variable $\hat{T}$.
  Count: $m_maxNM = O(m_maxNM)$

$$\sum_p \hat{T}_{s,l,m,p} \leq 1$$

- Memory type constraint: Each logical table has a *match type* that restricts the table's memory to a memory type. For example, a prefix IPv4 must be assigned to TCAM (ternary, wildcard memory), whereas an exact-match MAC table can be assigned to either SRAM (exact memory) or TCAM. The preprocessor returns, $f_{\mathbb{1}}(l, m)$, an indicator if table l can be assigned to memory m.
  Count: $m_{max}N = O(m_{max}N)$

$$\sum_s U_{s,l,m} \leq (b_m M) \cdot f_{\mathbb{1}}((l, m)$$

- Action memory constraint: There must at least be as many action words as match words for each logical table in each stage.
  Count: $O(NM)$

$$W_{s,l,a} \geq \sum_m W_{s,l,m} \qquad \text{// if table } l \text{ needs action memory}$$

Binary constraints

It takes a lot to get binary constraints working. With the exception of the objective function boolean variables, all boolean constraints are located in this section. Objective function-related constraints are located in their respective sections.

- Start and end stage binary per logical table (for dependency constraints).
  Count: $6NM = O(NM)$

$$P_{s,l}^{\hat{S},\hat{\tilde{U}}} \leq \hat{S}_{s,l} \qquad P_{s,l}^{\hat{E},\hat{\tilde{U}}} \leq \hat{E}_{s,l}$$
$$P_{s,l}^{\hat{S},\hat{\tilde{U}}} \leq \hat{\tilde{U}}_{s,l} \qquad P_{s,l}^{\hat{E},\hat{\tilde{U}}} \leq \hat{\tilde{U}}_{s,l}$$
$$P_{s,l}^{\hat{S},\hat{\tilde{U}}} \geq \hat{S}_{s,l} + \hat{\tilde{U}}_{s,l} - 1 \qquad P_{s,l}^{\hat{E},\hat{\tilde{U}}} \geq \hat{E}_{s,l} + \hat{\tilde{U}}_{s,l} - 1$$

- Packing unit binary. The number of packing units $T_{s,l,m,p}$ is in the range $[0, M]$.
  Count: $2(m_{max}NMp_{max}) + 2NM = O(m_{max}NMp_{max})$

$$\hat{T}_{s,l,m,p} \leq T_{s,l,m,p} \text{ (LB)}, \qquad \hat{T}_{s,l,m,p} \cdot M \geq T_{s,l,m,p} \text{ (UB)} \qquad \text{Match memory}$$
$$\hat{T}_{s,l,a,1} \leq T_{s,l,a,1} \text{ (LB)}, \qquad \hat{T}_{s,l,m,a} \cdot M \geq T_{s,l,m,1} \text{ (UB)} \qquad \text{Action memory}$$

- Block binary.
  Count: $2NM = O(NM)$

$$\hat{U}_{s,l,m} \leq U_{s,l,m} \text{ (LB)}, \qquad \hat{U}_{s,l,m} \cdot M \geq U_{s,l,m} \text{ (UB)}$$

- Aggregation block binary.
  Count: $2NM = O(NM)$

$$\hat{\tilde{U}}_{s,l} \leq \sum_m U_{s,l,m} \text{ (LB)}, \qquad (\sum_m b_m)\hat{\tilde{U}}_{s,l} \geq \sum_m U_{s,l,m} \text{ (UB)}$$

Common constraints

These are the constraints in Section 4.1 of the attached paper, which are the constraints required of any program assignment to any match-action switch.

- Capacity constraint. b_m is the number of blocks of memory type m in each stage.
  Count: $m_{max}NM + NM = O(m_{max}NM)$
  - General constraint for match memory.

$$\sum_l U_{s,l,m} \leq b_m$$

  - Constraint for SRAM ($m = 1$), which must also include action memory.

$$\sum_l (U_{s,l,1} + U_{s,l,a}) \leq b_1$$

- Assignment constraint. e_l is the number of entries of table l.
  Count: $N = O(N)$

$$\sum_m \sum_s W_{s,l,m} \geq e_l.$$

- Dependency constraint. This involves assigning all of the dependency variables as well, where ϵ is some small non-zero margin.
  Count: $2N + 2NM + 2N + d_{tot} = O(NM + d_{tot})$, where d_{tot} is the total number of match, action, successor, and reverse match dependencies.
  - Unique start and end stage constraints. There is a unique start and end stage for each logical table l. These constraints allow us to get the start and end stage for table l, $\sum_s (\hat{S}_{s,l} \cdot s)$ and $\sum_s (\hat{E}_{s,l} \cdot s)$, respectively.

$$\sum_s \hat{S}_{s,l} == 1$$

$$\sum_s \hat{E}_{s,l} == 1$$

  - Start and end stage constraint. If a stage has blocks, the start stage is at least as small, and the end stage is at least as big. M is an upper bound.

$$\sum_{s'} (\hat{S}_{s',l} \cdot s') \leq s + (1 - \hat{\hat{U}}_{s,l})M$$

$$\sum_{s'} (\hat{E}_{s',l} \cdot s') \geq s + (1 - \hat{\hat{U}}_{s,l})M$$

– Non-zero start and stages. Both the start and end stages must have blocks.

$$\sum_s P_{s,l}^{\hat{S},\hat{\tilde{U}}} \geq 1$$

$$\sum_s P_{s,l}^{\hat{E},\hat{\tilde{U}}} \geq 1$$

– Match and action dependencies. Suppose table l_2 has a match or action dependency on l_1, so l_2 must execute after l_1.

$$\sum_s (\hat{S}_{s,l_2} \cdot s) \geq \sum_s (\hat{E}_{s,l_1} \cdot s) + \epsilon$$

– Successor and reverse match dependencies. Suppose table l_2 has a successor or reverse dependency on l_1.

$$\sum_s (\hat{S}_{s,l_2} \cdot s) \geq \sum_s (\hat{E}_{s,l_1} \cdot s)$$

RMT-specific constraints

- Match crossbar constraint. The number of match crossbar units needed per table l for memory m is computed by the preprocesor as $f_{match}(l,m)$ as a function of the table width. The maximum match subunits for memory m is a constant, $c_{match,m}$.
  Count: $m_{max}M = O(m_{max}M)$

$$\sum_l (\hat{U}_{s,l,m} f_{match}(l,m)) \leq c_{match,m}$$

- Maximum number of tables per stage constraint. The maximum number of tables that can be matched per stage is a constant, τ, and this is constant across stages in our model.
  Count: $M = O(M)$

$$\sum_l \hat{\tilde{U}}_{s,l} \leq \tau$$

- Action crossbar constraint. The number of action crossbar units needed per table l for memory m is computed by the preprocesor as $f_{action}(l,m)$ as a function of the table width. The maximum action subunits for memory m is a constant, $c_{action,m}$.
  Count: $m_{max}M = O(m_{max}M)$

$$\sum_l (\hat{U}_{s,l,m} f_{action}(l,m)) \leq c_{action,m}$$

Maximum stage constraints

Equation 4 in Section 4.1 of the attached paper is recreated as follows. These constraints allow us to get the maximum stage, $\sum_s(\hat{\sigma}_s \cdot s)$.
Count: $2M + 3M + 1 + M + M = O(M)$

- Binary constraints. $\hat{B}_s$ is an indicator of whether there are any blocks assigned across all memories and all tables, not including action memory.

$$\hat{B}_s \leq \sum_m \sum_l U_{s,l,m} \text{ (LB)}, \qquad (\sum_m b_m)\hat{B}_s \geq \sum_m \sum_l U_{s,l,m} \text{ (UB)}$$

- Product constraints.

$$P_s^{\hat{\sigma},\hat{B}} \leq \hat{B}_s$$
$$P_s^{\hat{\sigma},\hat{B}} \leq \hat{\sigma}_s$$
$$P_s^{\hat{\sigma},\hat{B}} \leq \hat{B}_s + \hat{\sigma}_s - 1$$

- Unique maximum stage constraint.

$$\sum_s \hat{\sigma}_s == 1$$

- Maximum stage constraint. If a stage has blocks, the maximum stage is at least as big.

$$\hat{B}_s \cdot s \leq \sum_{s'} (\hat{\sigma}_{s'} \cdot s')$$

- Non-zero maximum stage. The maximum stage must have at least one block. This removes the possibility of the maximum stage being 0.

$$\sum_s P_s^{\hat{\sigma},\hat{B}} > 0$$

Pipeline latency constraints

Count: $6NM + (M-1) + (d_{match} + d_{action}) = O(NM + M + d_{tot})$, where $d_{match}, d_{action}, d_{tot}$ are the number of match, action, and total dependencies, respectively.

- Product resolution of stage starting times t_s and start/end stage dependency variables $\hat{S}_{s,l}$ and $\hat{S}_{s,l}$. The upper bound on $P_{s,l}^{\hat{S},t}$ is the product of the upper bound on t_s and on $\hat{S}_{s,l}$, or $(12M)M = 12M^2$, where 12 cycles is the maximum execution time per stage.

$$P_{s,l}^{\hat{S},t} \leq \hat{S}_{s,l}(12M^2) \qquad P_{s,l}^{\hat{E},t} \leq \hat{E}_{s,l}(12M^2)$$
$$P_{s,l}^{\hat{S},t} \leq t_s + (1 - \hat{S}_{s,l})(12M^2) \qquad P_{s,l}^{\hat{E},t} \leq t_s + (1 - \hat{E}_{s,l})(12M^2)$$
$$P_{s,l}^{\hat{S},t} \geq t_s + (1 - \hat{S}_{s,l})(12M^2) \qquad P_{s,l}^{\hat{E},t} \geq t_s + (1 - \hat{E}_{s,l})(12M^2)$$

- General latency constraint between stages. Regardless of dependencies, successive stages must be separated by at least one cycle.

$$\sum_s \hat{S}_{s,l} \geq \sum_{s-1} \hat{E}_{s-1,l} + 1$$

- Match latency constraint. Successive stages containing tables with match dependencies must be separated by 12 cycles. Table l_2 has a match dependency on table l_1. The start times of the start and end stages of table l are $\sum_s P_{s,l}^{\hat{S},t}$ and $\sum_s P_{s,l}^{\hat{E},t}$, respectively.

$$\sum_s P_{s,l_2}^{\hat{S},t} \geq \sum_s P_{s,l_1}^{\hat{E},t} + 12$$

- Action latency constraint. Successive stages containing tables with action dependencies must be separated by 3 cycles. Table l_2 has an action dependency on table l_1.

$$\sum_s P_{s,l_2}^{\hat{S},t} \geq \sum_s P_{s,l_1}^{\hat{E},t} + 3$$

Objectives

- Maximum stage.

$$\min \sum_s (\hat{B}_s \cdot s)$$

- Pipeline latency. Minimize the starting time of the last stage, t_M.

$$\min t_M$$

- Power. f_l is the width in bits of table l's entries, and w_m is the width in bits of memory type m. g_m is the watts per memory block of type m. We compute B_m, the number of active blocks of memory type m and take a weighted sum. Active SRAM blocks include match and action memory. The active TCAM blocks is weighted by the fraction of the bits used in each block.

$$B_1 = \sum_s \sum_l (U_{s,l,1} + U_{s,l,a})$$

$$B_2 = \sum_s \sum_l \sum_p (T_{s,l,m,p} \cdot f_U(T,s,l,m,p) \cdot \frac{w_m}{f_l}) = \sum_s \sum_l (U_{s,l,m} \cdot \frac{w_m}{f_l})$$

$$\min \sum_m g_m \cdot B_m$$