Batched Vectorized Earley Parsing

Celine Lee, Alexander Rush
Cornell University

Vectorized Earley Parsing enables parallelization

1. Vectorized: algorithm with matrices means the efficiency of this system scales with improved vectorized hardware (GPUs).

2. Parallelized: steps of the Earley algorithm are run in parallel using matrix operations.

3. Batched: vectorized implementation enables parallelization along the batch dimension.

4. Earley algorithm can parse any context-free language, making it a useful tool for interpretable language applications (comp. ling., neurosym., etc.)

Task: parse B input strings $w^{1:B}$ with length l, given some parameterized grammar $G=\langle N,T,(S,S'),R,\theta\rangle$

Earley Parsing: as reading input string left-to-right, process states [rule dotstate ($A \rightarrow \alpha \bullet \beta$), start index ($i$), end index ($j$)] by complete, predict, or scan.

Serial Earley Parsing

$O(l^3|G.R|^2)$

```
for w ∈ B do
    E = [{}_{ordered,−i}] of length l + 1
    I_root = (S → ·S′, V = θ_{S→S′}, i = 0)
    E[0] ←∪ I_root
    for j = 0, ..., l do
        for I ∈ E[j] do
            if I is finished then                          ▷ Completion
                for I_s ∈ E[I.i] do
                    if I.nonterm=I_s.next symbol then
                        I_p ← I_s.advance
                        I_p.V ←⊕ I_s.V ⊗ I.V
            else
                a ← S.next symbol
                if a ∈ G.N then                            ▷ Prediction
                    for a → γ ∈ G do
                        E[j] ← (a → ·γ, V = θ_{a→γ}, i = j)
                else if a = w_j then                       ▷ Scanning
                    I_n ← I.advance
                    I_n.V ←⊕ I.V
    yield E[len(w)][S → S′·, i = 0].V
```

Vectorized Earley Parsing

Perform Earley as a series of matrix operations along dotstate (D) and start- and end- index dimensions. Support matrices encode properties of the grammar relevant to Earley Parsing. **All matrices are sparse.**

Implemented using sparse matrices, Batched Vectorized Earley parsing can be efficiently executed on vectorized hardware.

Initialization Matrix : $I \in [0,1]^{|D|}$. $I[d] = \theta_{A \rightarrow \alpha}$ iff $d \sim^D A \rightarrow \bullet \alpha$

Transition Matrix : $T \in \{0,1\}^{|N \cup T| \times |D| \times |D|}$. $T[s,d_a,d_b] = 1$ iff $d_a \sim^D A \rightarrow \alpha \bullet s\beta$ and $d_b \sim^D A \rightarrow \alpha s \bullet \beta$, where $s \in N \cup T$

Completion Matrix : $C \in \{0,1\}^{|D| \times |D|}$. $C[d_a,d_b] = 1$ iff $d_a \sim^D A \rightarrow \alpha \bullet$ and $d_b \sim^D B \rightarrow \alpha \bullet A\beta$

Support matrices help manipulate Earley Chart : $E \in [0,1]^{B \times l \times l \times |D|}$, indexed by batch entry, end index, start index, then dotstate.

Predict: $E[:,i,i,:] = I$

Predict all starting states with the same start and end index.

Scan: $E[:,j+1,:,:] \oplus= E[:,j,:,:] \; @ \; T[w_j]$

$E[:,j,:,:]$: states with end index j

$@ \; T[w_j]$: for those with w_j as the next token; copy scores into the dot-progressed state in the next time step

$\oplus\Rightarrow$: and accumulate into existing values.

Complete: $E[:,j,:,:] \oplus= ((E[:,j,i,:] \; @ \; C) \times E[:,i,:,:]) \; @ \; T.sum(0)$

$E[:,j,i,:]$: states with start index i and end index j

$@ \; C$: grouped and combined by source nonterminal, indexed by items with that nonterminal next after the dot

$* \; E[:,i,:,:]$: multiply with states with end index i to get joint scores of completed items, indexed by pre-dot-progressed result state

$@ \; T.sum(0)$: advance to result dotstate

$\oplus\Rightarrow$: and accumulate into existing values.

Return: $E[:,l,0,d_f]$

Return the values associated with the final state [$(S \rightarrow S'\bullet)$, $i=0$, $j=l$].

```
E[:, diagonal, :] ← I                                ▷ Prediction
for j = 0, ..., l do
    for i = j, ..., 0 do                             ▷ Completion
        v ← E[:, j, i]@C
        v ← v.reshape(B, l, |D|) ⊗ E[:, i]
        E[:, j] ←⊕ v@T.sum(0)
    E[:, j + 1] ←⊕ E[:, j]@T[w_j]                     ▷ Scanning
return E[:, l, 0, d_f]
```

Batched Vectorized Earley fully parallelizes along the batch dimension and partially parallelizes along the start index loop.

Earley Parsing Properties

States must obey the **partial order relation** ≤ : If $A \le B$ then B must be processed after A is processed. States within the same partial order can be processed in parallel.

$$A \le B \text{ iff either: } \begin{cases} j_A < j_B \\ j_A = j_B \text{ and } i_A > i_B \end{cases}$$

This partial order must be maintained for Earley parsing implementations to be correct.

Processing operations $\oplus$, $\otimes$, @ can be implemented according to any parsing semiring* (inside, Viterbi, derivation forest, etc.), and the algorithm remains the same.

Parsing can be conducted in any semiring.

Results

Simple grammar (13 rules); inputs length ~ 5

	Viterbi		Inside	
	Serial	Vectorize	Serial	Vectorize
B = 2	0.0045 s	0.00414 s	0.0010 s	0.0030 s
B = 8	0.0182 s	0.0079 s	0.0040 s	0.0043 s

Impact

Earley parsing can be used for any grammar-based processes. Vectorizing Earley parsing enables Earley parsing over larger grammars and inputs as the capabilities of vectorized hardware machines scale.

Batched Vectorized Earley Parsing: Celine Lee, Alexander Rush, 10th Mid-Atlantic Student Colloquium on Speech, Language, and Learning. George Mason University, Arlington Campus, Virginia. 2023

* Semiring Parsing (Goodman, 1999)