In this paper, we address the task of parallel web page mining by using hyperlink information of web pages within bilingual websites.

We propose an iterative link-based approach which combines both internal and external translation similarity of web pages to identify parallel web pages.

### Motivation

#### Enhanced Translation Similarity

\[ ETS(e, c) = (1 - \alpha) \cdot S\text{in}\{e, c\} + \alpha \cdot S\text{ext}\{e, c\}, \alpha \in [0, 1] \]

#### Internal Translation Similarity

\[ S\text{in}\{e, c\} = \beta \cdot S\text{cb}\{e, c\} + (1 - \beta) \cdot S\text{struct}\{e, c\}, \beta \in [0, 1] \]

#### External Translation Similarity

\[ S\text{ext}\{e, c\} = \text{Sim}(PG\{e\}, PG\{c\}) \]

The solid arrows are the links between pages within a bilingual website. The dashed arrows indicate that page pair has high translation similarity.

We hypothesize that pages < C, C’ > might be parallel web pages if page C’s neighbors {A, B, D, E} have a higher translation similarity with page C’ ‘s neighbors {A’, B’, D’, E’} respectively.

### Algorithm Flow

#### Algorithm 1: Estimating the external translation similarity

**Input:** \( PG\{e\}, PG\{c\} \)

**Output:** \( S\text{ext}\{e, c\} \)

**Procedure:**

1. **Initialization:**
   - Set \( ETS\{e, c\} \) random value or small value

2. **LOOP:**
   - For each \( e \in P\{c\} \)
   - For each \( c \in P\{e\} \)
   - \( ETS\{e, c\} = \alpha \cdot S\text{struct}\{e, c\} + (1 - \alpha) \cdot S\text{in}\{e, c\} \)

**Parameters normalization:**

\[ S\text{ext}\{e, c\} = \text{Sim}(PG\{e\}, PG\{c\}) = 2 \cdot \text{Sim}(PG\{e\}, PG\{c\}) / (|PG\{e\}| + |PG\{c\}|) \]

#### Algorithm 2: Enhancing the internal translation similarity

**Input:** \( P\{e\}, P\{c\} \) (the English and Chinese page set)

**Output:** \( ETS\{e, c\} \), \( e \in P\{c\} \), \( c \in P\{e\} \)

**Procedure:**

1. **Initialization:**
   - Set \( ETS\{e, c\} \) random value or small value

2. **LOOP:**
   - For each \( e \in P\{c\} \)
   - For each \( c \in P\{e\} \)
   - \( ETS\{e, c\} = \alpha \cdot S\text{struct}\{e, c\} + (1 - \alpha) \cdot S\text{in}\{e, c\} \)

**Parameters normalization:**

\[ S\text{in}\{e, c\} = \text{Sim}(PG\{e\}, PG\{c\}) = 2 \cdot \text{Sim}(PG\{e\}, PG\{c\}) / (|PG\{e\}| + |PG\{c\}|) \]

#### Algorithm 3: Finding parallel page pairs

**Input:** \( P\{e\}, P\{c\} \), \( ETS\{x, y\}, x \in P\{c\}, y \in P\{c\} \), \( MAX\ P \) or \( MIN\ SIM\ )

**Output:** Parallel Page Pairs List : \( P\{\} \)

**Procedure:**

1. **LOOP:**
   - For each \( x \in P\{c\} \)
   - For each \( y \in P\{c\} \)
   - \( ETS\{x, y\} = \alpha \cdot S\text{struct}\{x, y\} + (1 - \alpha) \cdot S\text{in}\{x, y\} \)

**Parameters normalization:**

\[ S\text{in}\{x, y\} = \text{Sim}(PG\{x\}, PG\{y\}) = 2 \cdot \text{Sim}(PG\{x\}, PG\{y\}) / (|PG\{x\}| + |PG\{y\}|) \]

The baseline system only adopts internal information of web pages for identifying parallel page pairs. Figure 2 shows that when \( \beta \) is set to 0.6, the baseline system achieves the best performance. Thus, we always set \( \beta = 0.6 \).

The experimental results show that the external information of web pages is an effective feature to prune parallel web pages.

The input \( MAX\ P \) is an integer threshold which means that only top \( MAX\ P \) page pairs will be extracted in a certain website.

Figure 3 shows that the performance of our method achieves the maximal values and converges after the third iteration. In addition, Figure 3 indicates that our method is robust for different websites. Thus, the iteration number is set to 3 in the following experiments.