The Balanced Sliding Window Protocol
Definitions

- Two processes, \( p \) and \( q \), each sending an infinite array of words to the other

- **For Process** \( p \):
  
  \( \text{in}_p \):  An infinite array of words to be sent to process \( q \)

  \( \text{out}_p \): An infinite array of words being received from process \( q \)

  \( \text{Initially for all } i, \text{out}_p[i] = udef \)

  \( S_p \): The lowest numbered word that \( p \) still expects to receive from \( q \)

  At any time, \( p \) has already written \( \text{out}_p[0] \) through \( \text{out}_p[S_p - 1] \)
Required Properties

Safe delivery:

- In every reachable configuration of the protocol
  
  \[
  \text{out}_p[0 \ldots s_p - 1] = \text{in}_q[0 \ldots s_p - 1] \text{ and } \text{out}_q[0 \ldots s_q - 1] = \text{in}_p[0 \ldots s_q - 1]
  \]

Eventual delivery:

- For every integer \( k \geq 0 \), a configuration with \( s_p \geq k \) and \( s_q \geq k \) is eventually reached
The protocol

- The packet, \(<\text{pack}, w, i>\), transmits the word \(w = in_p[i]\) to \(q\).

- The processes use constants \(l_p\) and \(l_q\) as follows:
  - Process \(p\) can send the word \(w = in_p[i]\) (as the packet, \(<\text{pack}, w, i>\)) only after storing all the words \(out_p[0]\) through \(out_p[i-l_p]\), that is, \(i < s_p + l_p\).
  - When \(p\) receives \(<\text{pack}, w, i>\), retransmission of words from \(in_p[0]\) through \(in_p[i-l_q]\) is no longer necessary.
The Sliding Windows

\[ a_p \quad s_p + l_p \quad in_p \] 

\[ Q_q \quad p \quad Q_p \quad q \] 

\[ a_q \quad s_q + l_q \quad in_q \] 

\[ out_p \quad WWWWWWWu uRRu \quad s_p \] 

\[ WWWWWu uRRu u u \quad out_q \quad s_q \]
The Protocol

$S_p: \{ \ a_p \leq i < s_p + l_p \ \}$$
begin$$send < pack, in_p[i], i > to q$$end
$R_p: \{ < pack, w, i > \in Q_p \}$$
begin$$receive < pack, w, i > ;$$
if $out_p[i] = udef$ then$$begin$$out_p[i] = w ;$$
$$a_p = \max \{ a_p, i - l_q + 1 \};$$
$$s_p = \min \{ j \mid out_p[j] = udef \}$$
end$$// else ignore – retransmission$$end
$L_p: \{ < pack, w, i > \in Q_p \}$$
begin$$Q_p = Q_p \setminus \{ < pack, w, i > \}$$end
Protocol Invariant

\[ P \equiv \forall i < s_p : out_p[i] \neq udef \wedge \forall i < s_q : out_q[i] \neq udef \]
\[ \wedge < \text{pack, } w, i > \in Q_p \Rightarrow w = in_q[i] \wedge (i < s_q + l_q) \]
\[ \wedge < \text{pack, } w, i > \in Q_q \Rightarrow w = in_p[i] \wedge (i < s_p + l_p) \]
\[ \wedge out_p[i] \neq udef \Rightarrow out_p[i] = in_q[i] \wedge (a_p > i - l_q) \]
\[ \wedge out_q[i] \neq udef \Rightarrow out_q[i] = in_p[i] \wedge (a_q > i - l_p) \]
\[ \wedge a_p \leq s_q \]
\[ \wedge a_q \leq s_p \]
Results

**Safety:** The protocol satisfies the requirement of safe delivery

**Liveness:**

- $P$ implies $s_p - l_q \leq a_p \leq s_q \leq a_q + l_p \leq s_p + l_p$
- $P$ implies that the sending of $<\text{pack}, \text{in}_p[s_q], s_q>$ by $p$ or the sending of $<\text{pack}, \text{in}_q[s_p], s_p>$ by $q$ is applicable.
  - *Hence no deadlock is possible*
- The protocol satisfies the requirement of eventual delivery