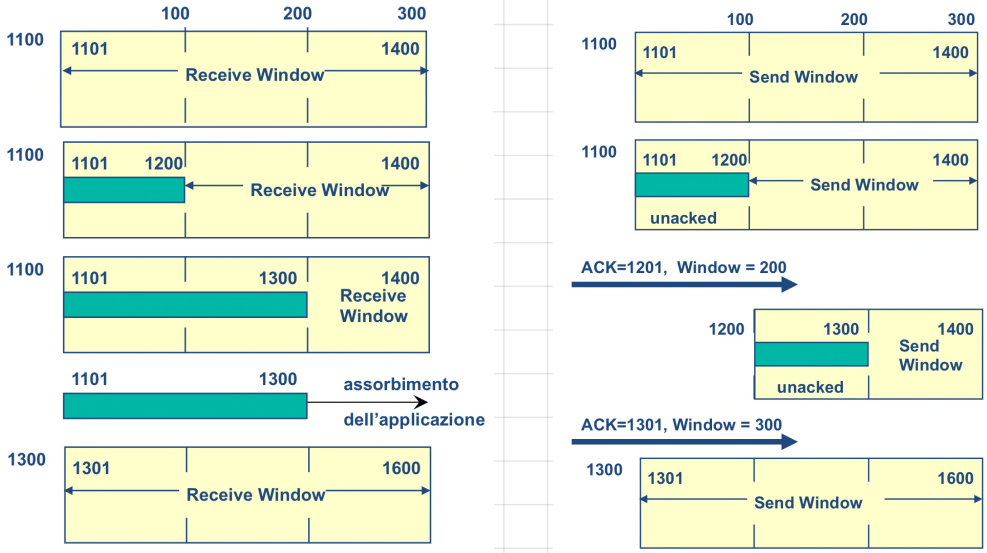
...

4.2.3.3 CONTROLLO FLUSSO

Il TCP prevede un buffer in ricezione e in invio. Lo spazio libero nel buffer del ricevitore viene detta Receive Window (RCVWND). La dimensione della finestra di ricezione è segnalata in ogni segmento.

I byte nella receive window sono numerati e riflettono l'ordine in cui essi devono essere ricevuti.

Analogamente la Send Window rappresenta i byte che possono essere trasmessi senza attendere riscontri. La send window si estende dal primo byte non riscontrato fino alla fine della receive window del ricevente



Le finestre hanno dei problemi. Uno di questi è la silly window syndrome. Essa è avviene lato ricevitore ed è causata da un ricevitore svuota il buffer troppo lentamente e quindi il buffer si riempie. Appena la finestra si svuota di poco il trasmettitore invia pochi byte con molto overhead. La soluzione è l'algoritmo di Clarke: il ricevente segnala finestra nulla finché essa non è abbastanza grande da ricevere un segmento di dimensione massima.

La S.W.S avviene anche a lato trasmettitore quando dati lentamente e, quindi, vengono prodotti pacchetti piccoli. La soluzione è l'algoritmo di Nagle: la prima porzione viene mandata corta, il resto solo se si riempie un segmento di dimensione massima.

I comportamenti sopra possono essere evitati usando il flag PUSH: esso invia i dati anche se non si riempie un segmento massimo

4.2.3.4 CONTROLLO ERRORE

Il meccanismo è di tipo GBN-TIMEOUT con alcune modifiche:

- i segmenti fuori ordine vengono mantenuti
- quando arrivano i segmenti mancanti la finestra scorre in avanti fino al primo pacchetto non riscontrato tra quelli fuori ordine
- viene mandato un ACK che riscontra collettivamente anche i segmenti fuori ordine.

4.2.3.5 GESTIONE TIMEOUT

Il RTT è molto volatile e, di conseguenza, il timeout va modificato per accomodare. Il TCP usa l'algoritmo di Karn e Jacobson.

- RTT viene definito come il tempo tra trasmissione e ricezione del relativo ACK
- sulla base dei campioni RTT il sender calcola il SmoothedRTT come: $SRTT^{(i)} = (1-\alpha)SRTT^{(i-1)} + \alpha RTT^{(i)}$ $(0 < \alpha < 1$, solito $\alpha = \frac{1}{8})$
- viene anche calcolata una variante smoothed della deviazione standard come: $SDEV^{(i)} = 3/4\ SDEV^{(i-1)} + 1/4\ DEV$
- viene calcolato il timeout grande. $T_O = SRTT + 4SDEV$

Il valore iniziale del timeout è 1s. A seguito di una ritrasmissione è meglio usare l'algoritmo di Karn

- RTT non viene aggiornato
- Il timeout è moltiplicato per un fattore fisso (tipicamente 2)
- Il *timeout* cresce fino ad un valore massimo
- Dopo un numero massimo di ritrasmissioni la connessione viene chiusa

4.2.3.6 PERSISTENZA

Se il ricevitore manda finestra pari a 0, la trasmissione si ferma finché il ricevitore non manda un ACK con la nuova window size diversa da 0. Per evitare blocaggio a causa di perdita di questo ACK viene usato un timer di persistenza. Se il timer di persistenza scade viene mandato un messaggio di probe. Se si riceve ACK, si fa riprendere la trasmissione, altrimenti si continua probare (ad ogni timeout del timer) finché non si riceve ACK.

4.2.3.7 CONTROLLO CONGESTIONE

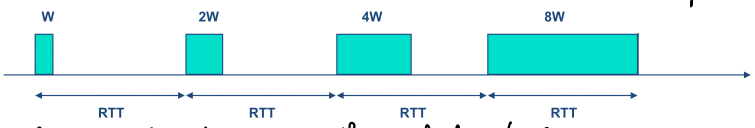
Il controllo congestione è di tipo end-to-end (la rete è, quindi, vulnerabile al sovraccarico in quanto ci si fida dell'host). La congestione viene regolata in base ai dati ricevuti (ACK, T.O ...) usando una Congestion Window (CWND) che viene regolata opportunamente. Il trasmettitore non può trasmettere più del minimo tra RCVWND e CWND

Per regolare la finestra di congestione si usano:

- Slow-Start (CWND ≤ SSTHRESH)
- Congestion avoidance (CWND > SSTHRESH)

La connessione parte in Slow-Start in quanto la finestra di congestione viene inizializzata a 1 MSS con SSTHRESH elevata.

Nello Slow-Start la CWND viene incrementata per ogni ACK. Il nome è a caso, perché è tutt'altro che lento.



Il slow-start rende esponenziale l'incremento di CWND. La velocità di trasmissione è pari a $R = \frac{CWND}{RTT}$

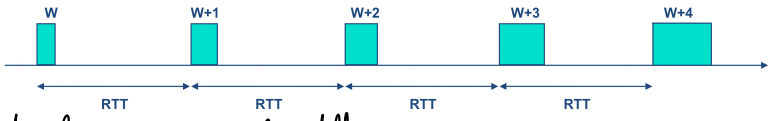
Un evento di congestione capita quando scade il timeout. Il TCP, in questi casi, reagisce modificando SSTHRESH con questa formula:

$$SSTHRESH = \max\left(2\,MSS, \frac{FLIGHT\_SIZE}{2}\right)$$

FLIGHT_SIZE byte trasmessi ma non riscontrati ≅ CWND

e rimettendo CWND a 1

La CONGESTION AVOIDANCE prevede l'incremento della CWND di 1/CWND ad ogni ACK ricevuto. Ciò significa che se la CWND consente di trasmettere N segmenti, allora la CWND aumenta di 1 segmento poiché riceverà N ACK



La CONGESTION AVOIDANCE rende lineare l'incremento della CWND.

Esempio di controllo della finestra: