TCP FACK
(Forward Acknowledment)

AGENDA

- Algoritmo de Recuperação Rápida
- TCP SACK
- TCP FACK
- Considerações Finais

Algoritmo de Recuperação Rápida

- Como uma evolução do Tahoe, o TCP Reno, incorpora o algoritmo **Fast Recovery** ao Fast Retransmit.

- **Fast Recovery** tem início quando:
  - Emissor receber 3 ACKs duplicados
  - Assim ocorre a retransmissão rápida
  - ssthresh = cwnd / 2
  - cwnd = ssthresh + 3 * MSS

Algoritmo de Recuperação Rápida

- Nessa estado:
  - A cada ACK duplicado recebido
    cwnd = cwnd + 1 * MSS

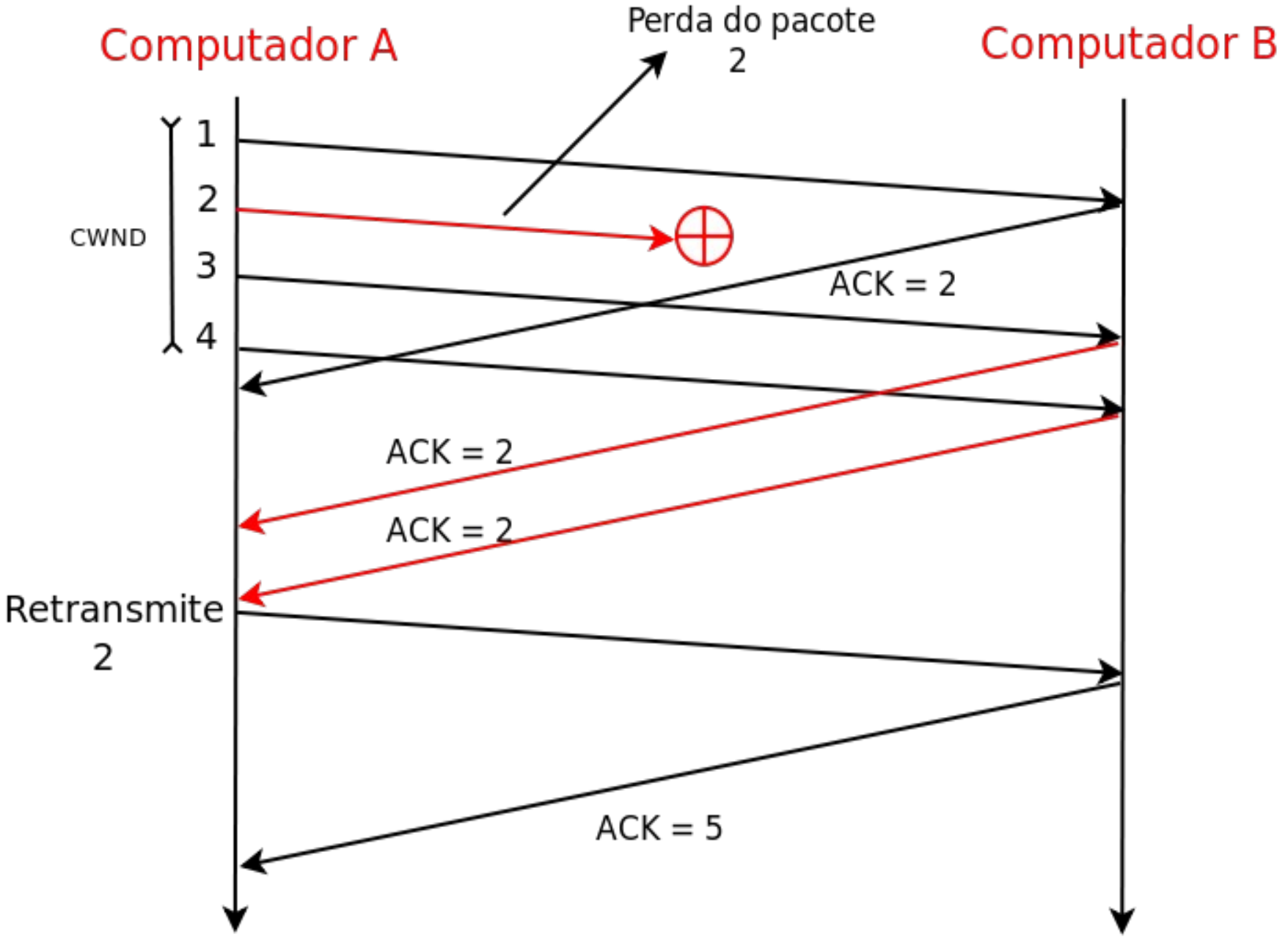
- O Estado termina:
  - Ao receber o novo ACK do pacote retransmitido. A variável **cwnd** fica sendo igual ao **ssthresh**.

- Após isso, transita-se para o estado de prevenção de congestionamento.

Algoritmo de Recuperação Rápida

- Se houver somente **uma perda** de pacote na janela, o recebimento do ACK do pacote retransmitido pode reconhecer todos os pacotes pertencentes à janela.

Algoritmo de Recuperação Rápida

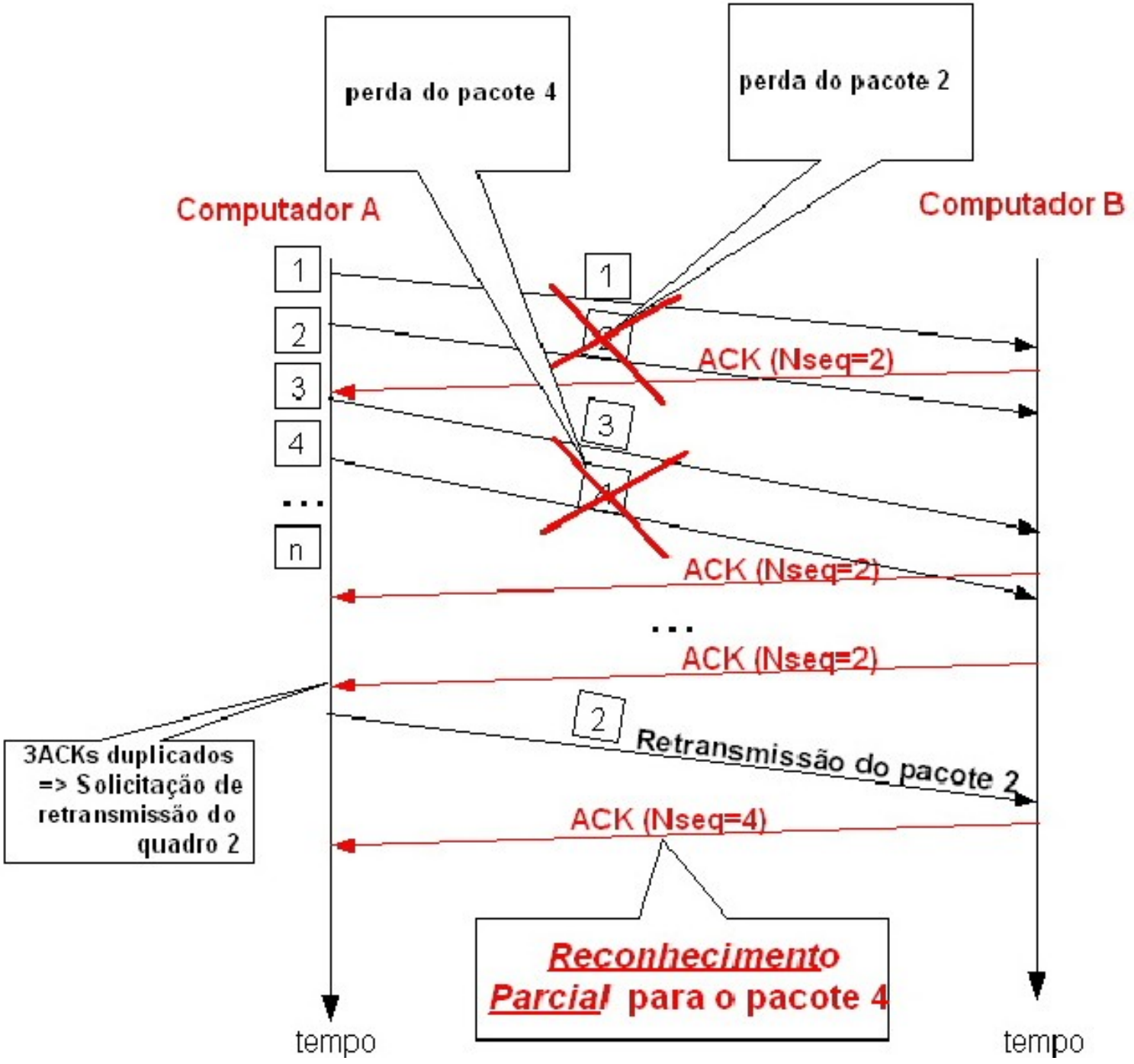


Algoritmo de Recuperação Rápida

- Entretanto, se houver **mais de uma perda** de pacotes na mesma janela, o *ACK* do pacote retransmitido reconhecerá todos os pacotes recebidos até a próxima perda.

- Este reconhecimento, que não confirma todos os pacotes enviados pertencentes à mesma janela é denominado **reconhecimento parcial**.

Reconhecimento Parcial



Algoritmo de Recuperação Rápida

- ***Delay*** na recuperação de múltiplos segmentos na mesma janela.

- A cada reconhecimento, o algoritmo **Fast Recovery** é acioando, dividindo a janela de congestionamento pela metade.

- Portanto, em presença de várias perdas de pacotes, o TCP Reno apresenta desempenho insatisfatório, em função das **sucessivas divisões** de cwnd.

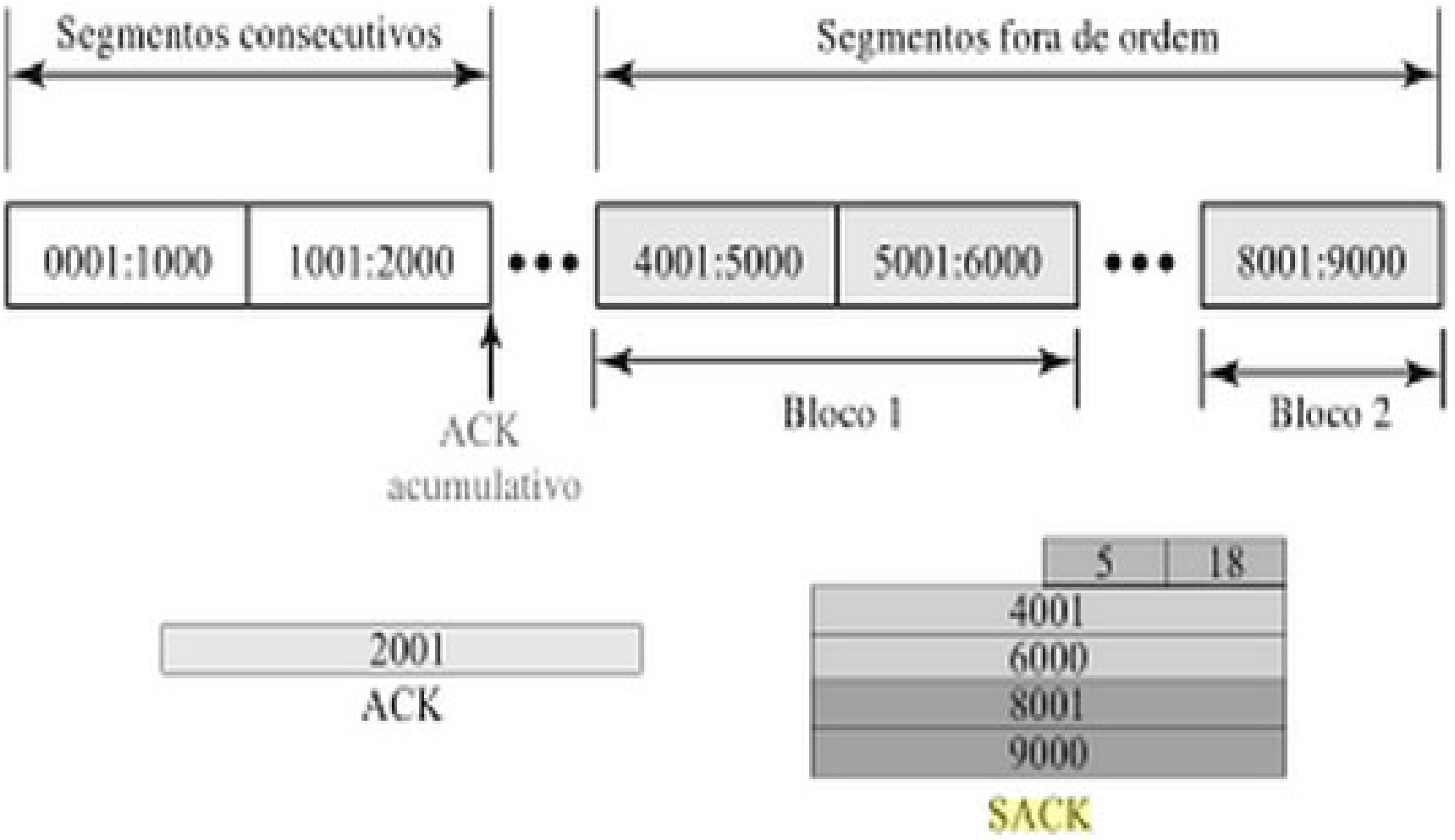
- Reconhecimento cumulativo do TCP só diz informação sobre pacotes recebidos

TCP SACK

- Visa otimizar o algoritmo de recuperação rápida utilizando um mecanismo de reconhecimento seletivo.

- Aproveita o campo **opções** do TCP para informar ao emissor sobre pacotes contínuos (blocos) que chegaram fora de ordem.

TCP SACK


Segmentos consecutivos
Segmentos fora de ordem
0001:1000
1001:2000
4001:5000
5001:6000
8001:9000
ACK
acumulativo
Bloco 1
Bloco 2
2001
ACK
5
18
4001
6000
8001
9000
SACK

TCP FACK

- Foi proposto em 1996 por **Matthew Mathis**.

- O FACK explora a informação fornecida pela **opção SACK** para promover um melhor controle da injeção de pacotes na rede durante a fase de recuperação dos dados.

- TCP FACK vai ter influência nesse dois elementos:
  - Controle de congestionamento
  - Recuperação Rápida

TCP FACK

- **Controle de Congestionamento**
  - Tenta estimar o total de pacotes em trânsito na rede.

  - Essa estimativa pode ser usada pelo emissor para decidir se envia ou não uma nova porção de dados.

- **Recuperação rápida**
  - Para múltiplas perdas em uma mesma janela o FACK verifica que há pacotes perdidos com mais rapidez.

TCP FACK
Controle de Congestionamento

- Estimativa da quantidade de dados em trânsito na rede é dado por:

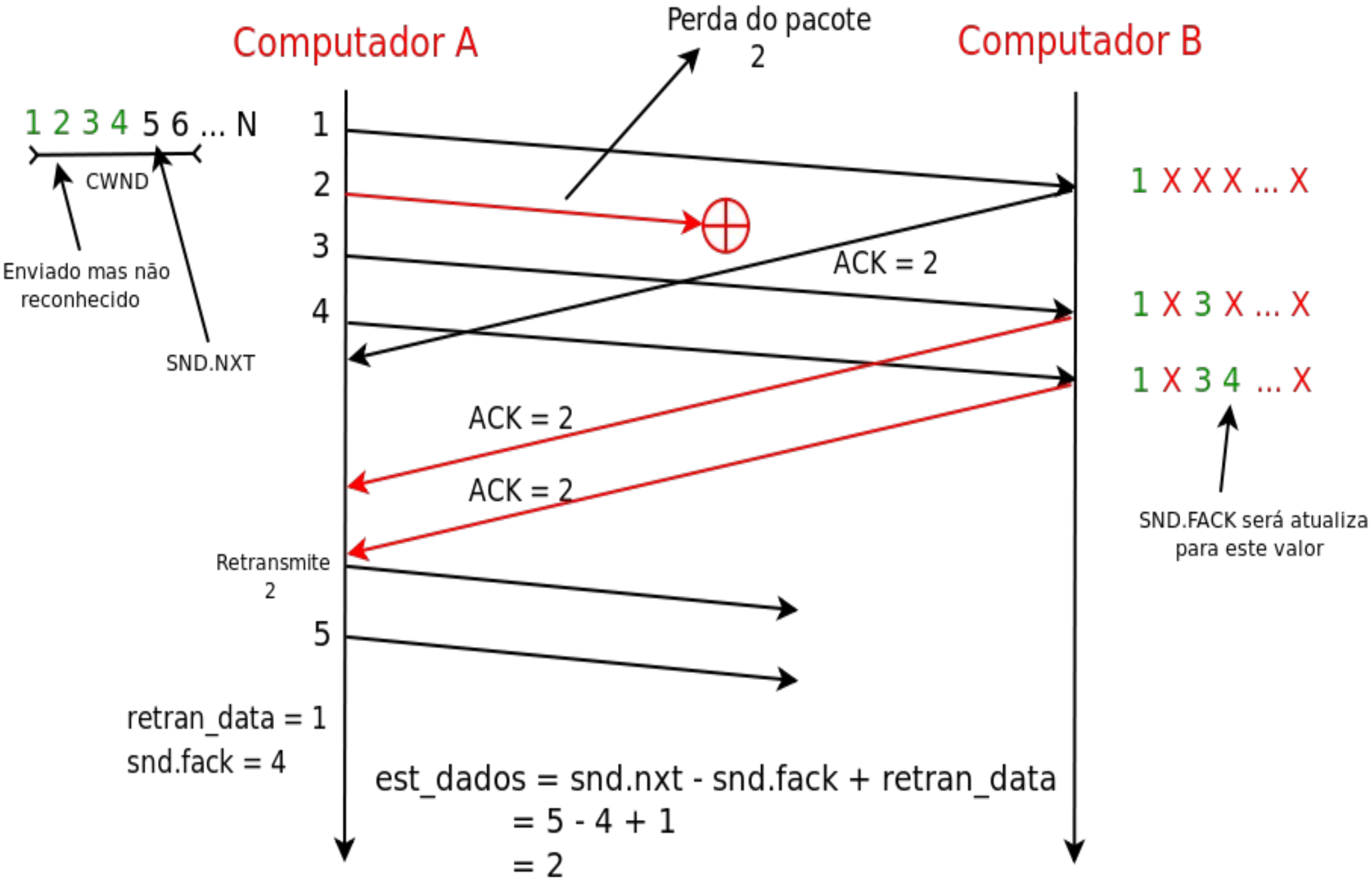
est_dados = snd.nxt – snd.f ack + retran_data

- **snd.nxt:** Número de sequência do próximo dado a ser transmitido

- **snd.fack:** Número de sequência do dado mais a frente recebido com sucesso no receptor. Atualizada pela opção SACK.

- **retran_data:** Quantidade de pacotes retransmitidos

Estimativa de pacotes em trânsito na rede



TCP FACK
Recuperação Rápida

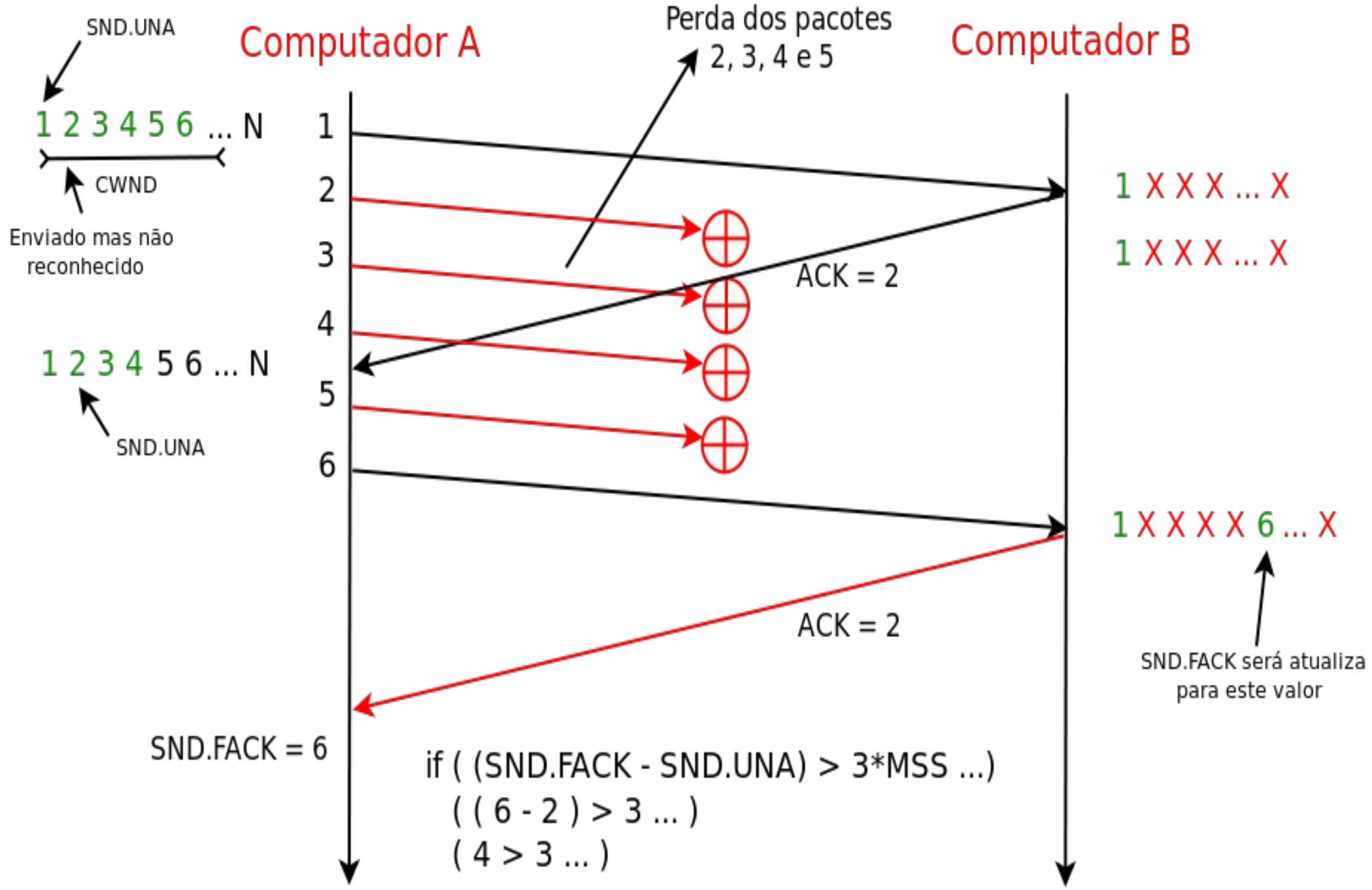
- Estima a identicação do pacote perdido

If((snd.fack – snd.una) > 3*MSS || dupacks == 3)

snd.una: Número de sequência do pacote enviado, mas ainda não reconhecido. Byte mais antigo.

dupacks: Acks duplicados.

TCP FACK
Recuperação Rápida



Considerações Finais

- TCP FACK proporciona melhora no desempenhho da conexão em relação a versoes anteriores do TCP, evitando-se retransmissoções acionadas pelo timeout.

- e pelo controle dos dados em trânsito na rede durante a fase de recuperação de dados perdidos.

- Melhor recuperação a múltiplas perdas em uma mesma janela.

Referências

- Forward Acknowledment: Refining TCP Congestion Contral. Matthew Mathis and Jamshid Mahdavi.

- Hos-to-Host Congestion Control for TCP. Alexandder Afanasyev, Neil Tilley, Peter Reither, and Leonard Kleinrock.

- Melhoria de Desempenho do protocolo TCP em canais de HF via escolha de parâmetros e emprego de técnicas de controle de erros. Márcio Dantas.