algoritmi bidirezionali

Filippo Magi

February 15, 2022

1 Algoritmo senza ottimizzazione

Algorithm 1 Ricerca del massimo flusso senza alcuna ottimizzazione

Require: Una rete (G, u, s, t) .
Ensure: valore del flusso massimo
  while TRUE **do**
    DoBfs(G)
    if t.flussoPassante= 0 **then**
      break
    end if
    sendFlow(t)
  end while
  return s.flussoUscente

Algorithm 2 Algoritmo DoBfs senza alcuna ottimizzazione

Require: rete (G, u, s, t)
Ensure: Ricerca del percorso di G e aggiornamento delle informazioni contenute in $N(G)$
  for all $n \in V(G)$ **do**
    n.Reset()
  end for
  $coda \leftarrow$ Coda di nodi
  $coda$.Enqueue(s)
  while $\neg coda.isEmpty$ **do**
    $element \leftarrow coda$.Dequeue()
    for all $edge \in \delta(element)$ **do**
      $n \leftarrow element.next$ {si fa notrae che con next qui si intende il nodo di e che non è $element$}
      if $n.flussoPassante = 0 \wedge ((u_t(edge) > 0 \wedge \neg e.reversed) \vee (f(edge) > 0 \wedge e.reversed))$ **then**
        {per il codice reversed è un oggetto diverso che, in fase di invio, va di nuovo cercato, per questo motivo ho optato di non usare più questa tipologia di archi}
        n.update$(edge, element)$
        if $edge.reversed$ **then**
          n.flussoPassante $\leftarrow \min(u_t(e), element.flussoPassante)$
        else
          n.flussoPassante $\leftarrow \min(f(e), element.flussoPassante)$
        end if
        if $n = t$ **then**
          return
        else
          $coda$.Enqueue(n)
        end if
      end if
    end for
  end while