```
    and etape6 ()=
      let ind=trouver_indice c t.(2* !i) in
      (
      match tcomp.(2* !i+1) with
          |0->t.(2* !i+1)<-c.((ind+1) mod n);
              tvois:=c.((ind+2) mod n)
          |_->t.(2* !i+1)<-c.((ind+n-1) mod n);
              tvois:=c.((ind+n-2) mod n)
      );
      tcomp.(2* !i+1)<-tcomp.(2* !i+1)+1;
```

Choix d'un arc du chemin initial

Test : le chemin obtenu est-il un cycle passant par tous les sommets ?

```
          if not (dejapris t (2* !i+1)) & est_un_tour c nouvc t !i n
(* La fonction est_un_tour en profite pour définir nouvc le nouveau chemin *)
              then
                  if est_mieux d !i G t
                  then
                      begin
                      copier nouvc c;
                      etape12 ()
                      end
                  else
                      begin
                      tcomp.(2* !i+2)<-0;
                      etape7 ()
                      end
              else
                  if tcomp.(2* !i+1)=2
                  then etape8 ()
                  else etape6 ()
```

Si le nouveau chemin est meilleur, on le garde.

```
    and etape7 ()=
          t.(2* !i+2)<-tcomp.(2* !i+2);
          tcomp.(2* !i+2)<-tcomp.(2* !i+2)+1;
          if not (dejapris t (2* !i+2)) & t.(2* !i+2)<> !tvois & !G-.d.(t.(2*
!i+1)).(t.(2* !i+2))>0.
          then
              begin
              tcomp.(2* !i+3)<-0;
              etape5 ()
              end
```

Choix d'un arc pour le nouveau chemin

```
          else
              if tcomp.(2* !i+2)=n
              then etape8 ()
              else etape7 ()
    and etape8 ()=
          if tcomp.(4)=n
          then etape9 ()
          else
              begin
              i:=1;
              etape7 ()
              end
    and etape9 ()=
          if tcomp.(3)=2
          then etape10 ()
          else
              begin
              i:=1;
              etape6 ()
              end
    and etape10 ()=
          if tcomp.(2)=n
          then etape11 ()
          else
              begin
              i:=0;
              etape4 ()
              end
    and etape11 ()=
          if tcomp.(1)=2
          then etape12 ()
          else
              begin
              i:=0;
              etape3 ()
              end
    and etape12 ()=
          if tcomp.(0)=n
          then c (* C'est la fin de l'algo *)
          else etape2 ()
    in etape2 ()
;;
```

Retours en arrière