

Corrigé type Examen de Algorithmique Avancée
et Complexité

Année 2020/2021 par Mr Haniche

Exercice 1

Partie 1

$$T_1(n) = \sum_{i=1}^n (1 + 2i) + c = n + c + 2 \sum_{i=1}^n i$$

$$= n + c + 2(n+1)n/2 = n + c + n^2 + n$$

$$= n^2 + 2n + c = \underline{\underline{O(n^2)}}$$

Partie 2

$$T_2(n) = \sum_{i=10}^{n-10} (1 + (i+10) - (i-10) + 5) * 2 + c$$

$$= c + \sum_{i=10}^{n-10} (1 + 25 * 2) = c + \sum_{i=10}^{n-10} 43$$

$$= c + (n-10-10+1) * 43$$

$$= 43n - 861 = \underline{\underline{O(n)}}$$

Partie 3

$$T_3(n) = 1 + \sum_{i=1}^n ((i+5) * c + 5) + 5 \text{ avec } i=1, \frac{n}{2}, \frac{n}{4}, \dots, \frac{n}{2^{\log_2 n}}$$

$$= c + \sum_{i=1}^n 3 + c \sum_{i=1}^n i \text{ avec } i=1, \frac{n}{2}, \frac{n}{4}, \dots, 1$$

$$= c + 3 \log_2 n + c \underbrace{\left(n + \frac{n}{2} + \frac{n}{4} + \dots + \frac{n}{2^{\log_2 n}} \right)}_S \text{ de valeur } = \log_2 n$$

calcul de S

$$S - \frac{1}{2}S = n - \frac{1}{2}n$$

$$\Rightarrow \frac{1}{2}S = n - \frac{1}{2}n \Rightarrow S = 2n - 1$$

car $\frac{1}{2}S = \frac{2}{2} + \frac{1}{2} + \dots + \frac{1}{2}$

$$\text{donc } T_3(n) = c + 3 \log_2 n + c(2n - 1)$$

$$= 4n + 3 \log_2 n = \underline{\underline{O(n)}}$$

Exercice n° 2

L'Algorithme se base sur la sélection du fichier de taille inférieur parmi les fichiers non choisis, et cela dans chaque itération du processus glouton.

La préparation de donnée implique :

- 1- une initialisation du tableau S à zéro
- 2- tri croissant, selon la taille des fichiers, du tableau P contenant les informations sur les fichiers, l'indice du tableau P sera utilisé comme identifiant du fichier de ce numéro.

Algorithme

const n = ...

var

P : tableau [1..n] réel;
 S : tableau [1..n] entier;
 L, Som : réel;

Début

< tri du tableau P >

Tri par fusion par exemple
 $O(n \log n)$

< initialisation de S >

Pour $i := 1$ à n faire $S[i] := 0$ fin $O(n)$

lire(L);

Som := 0; $i := 1$; $O(1)$

tant que $Som + P[i] < L$

faire

$S[i] := 1$;

$i := i + 1$;

tant;

$Som := Som + P[i]$

le pire des cas correspond à l'ajout de tout les fichiers c-à-d un nombre d'itérations = n
dans ce cas $O(n)$

fin.

La complexité totale

Exercice - Suite
M. Hamiche F.

remarque

si le tableau n'est pas trié
 On aura une complexité $O(n^2)$
 car dans la boucle pour chaque
 itération on aura besoin de rechercher
 le min dans le tableau qui contient
 $a_n + b$ et donc

$$T(n) = n \times (a_n + b) = O(n^2)$$

nbr d'itération
 coût de la recherche

il est possible de donner
 l'algorithme sous le schéma
 Vue en cours (mais puisque ce n'est
 pas compliqué, on a donné cette version
 simplifiée)

schéma Vue à Cours
 (idc) (a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l) (m) (n) (o) (p) (q) (r) (s) (t) (u) (v) (w) (x) (y) (z) (aa) (ab) (ac) (ad) (ae) (af) (ag) (ah) (ai) (aj) (ak) (al) (am) (an) (ao) (ap) (aq) (ar) (as) (at) (au) (av) (aw) (ax) (ay) (az) (ba) (bb) (bc) (bd) (be) (bf) (bg) (bh) (bi) (bj) (bk) (bl) (bm) (bn) (bo) (bp) (bq) (br) (bs) (bt) (bu) (bv) (bw) (bx) (by) (bz) (ca) (cb) (cc) (cd) (ce) (cf) (cg) (ch) (ci) (cj) (ck) (cl) (cm) (cn) (co) (cp) (cq) (cr) (cs) (ct) (cu) (cv) (cw) (cx) (cy) (cz) (da) (db) (dc) (dd) (de) (df) (dg) (dh) (di) (dj) (dk) (dl) (dm) (dn) (do) (dp) (dq) (dr) (ds) (dt) (du) (dv) (dw) (dx) (dy) (dz) (ea) (eb) (ec) (ed) (ee) (ef) (eg) (eh) (ei) (ej) (ek) (el) (em) (en) (eo) (ep) (eq) (er) (es) (et) (eu) (ev) (ew) (ex) (ey) (ez) (fa) (fb) (fc) (fd) (fe) (ff) (fg) (fh) (fi) (fj) (fk) (fl) (fm) (fn) (fo) (fp) (fq) (fr) (fs) (ft) (fu) (fv) (fw) (fx) (fy) (fz) (ga) (gb) (gc) (gd) (ge) (gf) (gg) (gh) (gi) (gj) (gk) (gl) (gm) (gn) (go) (gp) (gq) (gr) (gs) (gt) (gu) (gv) (gw) (gx) (gy) (gz) (ha) (hb) (hc) (hd) (he) (hf) (hg) (hh) (hi) (hj) (hk) (hl) (hm) (hn) (ho) (hp) (hq) (hr) (hs) (ht) (hu) (hv) (hw) (hx) (hy) (hz) (ia) (ib) (ic) (id) (ie) (if) (ig) (ih) (ii) (ij) (ik) (il) (im) (in) (io) (ip) (iq) (ir) (is) (it) (iu) (iv) (iw) (ix) (iy) (iz) (ja) (jb) (jc) (jd) (je) (jf) (jg) (jh) (ji) (jj) (jk) (jl) (jm) (jn) (jo) (jp) (jq) (jr) (js) (jt) (ju) (jv) (jw) (jx) (jy) (jz) (ka) (kb) (kc) (kd) (ke) (kf) (kg) (kh) (ki) (kj) (kk) (kl) (km) (kn) (ko) (kp) (kq) (kr) (ks) (kt) (ku) (kv) (kw) (kx) (ky) (kz) (la) (lb) (lc) (ld) (le) (lf) (lg) (lh) (li) (lj) (lk) (ll) (lm) (ln) (lo) (lp) (lq) (lr) (ls) (lt) (lu) (lv) (lw) (lx) (ly) (lz) (ma) (mb) (mc) (md) (me) (mf) (mg) (mh) (mi) (mj) (mk) (ml) (mm) (mn) (mo) (mp) (mq) (mr) (ms) (mt) (mu) (mv) (mw) (mx) (my) (mz) (na) (nb) (nc) (nd) (ne) (nf) (ng) (nh) (ni) (nj) (nk) (nl) (nm) (nn) (no) (np) (nq) (nr) (ns) (nt) (nu) (nv) (nw) (nx) (ny) (nz) (oa) (ob) (oc) (od) (oe) (of) (og) (oh) (oi) (oj) (ok) (ol) (om) (on) (oo) (op) (oq) (or) (os) (ot) (ou) (ov) (ow) (ox) (oy) (oz) (pa) (pb) (pc) (pd) (pe) (pf) (pg) (ph) (pi) (pj) (pk) (pl) (pm) (pn) (po) (pp) (pq) (pr) (ps) (pt) (pu) (pv) (pw) (px) (py) (pz) (qa) (qb) (qc) (qd) (qe) (qf) (qg) (qh) (qi) (qj) (qk) (ql) (qm) (qn) (qo) (qp) (qq) (qr) (qs) (qt) (qu) (qv) (qw) (qx) (qy) (qz) (ra) (rb) (rc) (rd) (re) (rf) (rg) (rh) (ri) (rj) (rk) (rl) (rm) (rn) (ro) (rp) (rq) (rr) (rs) (rt) (ru) (rv) (rw) (rx) (ry) (rz) (sa) (sb) (sc) (sd) (se) (sf) (sg) (sh) (si) (sj) (sk) (sl) (sm) (sn) (so) (sp) (sq) (sr) (ss) (st) (su) (sv) (sw) (sx) (sy) (sz) (ta) (tb) (tc) (td) (te) (tf) (tg) (th) (ti) (tj) (tk) (tl) (tm) (tn) (to) (tp) (tq) (tr) (ts) (tt) (tu) (tv) (tw) (tx) (ty) (tz) (ua) (ub) (uc) (ud) (ue) (uf) (ug) (uh) (ui) (uj) (uk) (ul) (um) (un) (uo) (up) (uq) (ur) (us) (ut) (uu) (uv) (uw) (ux) (uy) (uz) (va) (vb) (vc) (vd) (ve) (vf) (vg) (vh) (vi) (vj) (vk) (vl) (vm) (vn) (vo) (vp) (vq) (vr) (vs) (vt) (vu) (vv) (vw) (vx) (vy) (vz) (wa) (wb) (wc) (wd) (we) (wf) (wg) (wh) (wi) (wj) (wk) (wl) (wm) (wn) (wo) (wp) (wq) (wr) (ws) (wt) (wu) (wv) (ww) (wx) (wy) (wz) (xa) (xb) (xc) (xd) (xe) (xf) (xg) (xh) (xi) (xj) (xk) (xl) (xm) (xn) (xo) (xp) (xq) (xr) (xs) (xt) (xu) (xv) (xw) (xx) (xy) (xz) (ya) (yb) (yc) (yd) (ye) (yf) (yg) (yh) (yi) (yj) (yk) (yl) (ym) (yn) (yo) (yp) (yq) (yr) (ys) (yt) (yu) (yv) (yw) (yx) (yy) (yz) (za) (zb) (zc) (zd) (ze) (zf) (zg) (zh) (zi) (zj) (zk) (zl) (zm) (zn) (zo) (zp) (zq) (zr) (zs) (zt) (zu) (zv) (zw) (zx) (zy) (zz)

préparation de données

Tant que T complet()

$a_i = \text{select}(c)$

Ajouter(a);

fin();

Tri divisionnaire

Si $a_i = idc$
 $S_{end} := A$
 $S_{om} := S_{om} + P[a_i]$
 $idc := idc + b$

si $a_i < idc$
 si $a_i > idc$
 si $a_i = idc$

2) la version optimiser de l'ordre de $O(n \log n)$
 va se baser sur la technique diviser pour régner
 et l'Algorithme utilise des fonction récursive
 en cours
 Algorithme rech_sou_x

31

Début

Tri_fusion (T, n); $\sim O(n \log n)$

i := 1;

tant que i <= n et T tr

faire

b := x - T[i];

tr := Rech_dichotomique (T, i, n, b);

i := i + 1;

fin;

fin

$O(n \log n)$

donc

l'Algorithme et de Recherche dichotomique
 l'ordre de $O(n \log n)$ de b dans le tableau T entre
 la case i et la case n de l'ordre de $O(\log n)$

Exercice 3

3- Un algorithme naïf pour ce problème fait la recherche sur le tableau brut sans préparation à l'avance, et se base sur une recherche très simple de deux indices de table i, j tq $T[i] + T[j] = x$

Algorithme Rech_som_x

Début

lire (x)

Pour $i := 1$

$i := 1$

tant que $i \leq n$ et $T \neq \emptyset$

faire

$j := i$
 $b := x - T[i]$

tant que $j \leq n$ et $T \neq \emptyset$

faire

si $T[j] = b$

alors écrire (i, j)
 $tr := \text{vrai}$

sinon $j := j + 1$

fin

fin

fin; $i := i + 1$

et à l'itération

fin.

$$T(n) = 2 + \sum_{i=1}^n (2 + 5 + (n-i) \cdot 3)$$

$$= 2 + 3n^2 + 7n + 3 \sum_{i=1}^n i = 3n^2 + 7n + 2 + \frac{3}{2}(n^2 + n)$$

$$= \frac{3}{2}n^2 + \frac{11}{2}n + 2 = O(n^2)$$