Lancer une flopée de fils d'exécution en parallèle en C

Afin de lancer un grand nombre ♣de fils d'exécution en parallèle, il faut les déclarer non pas explicitement un à un, mais dans un tableau à l'aide d'un `malloc`. De même les structures qui servent à passer les arguments aux fils d'exécution doivent être stockées dans des tableaux. Il doit y avoir autant de structures différentes physiquement (placées à des endroits différents en mémoire) que de fils d'exécution. En particulier déclarer les arguments dans une structure qu'on modifie à chaque tour de boucle ne marche pas : l'espace alloué pour les arguments du premier fil est un espace dans la pile qui est modifié dès le deuxième tour de boucle, et potentiellement avant que le premier fil n'ait pu lire ses arguments, et/ou écrire son résultat. On donne ci-dessous un exemple type de lancement d'un grand nombre de fils d'exécution.

```c
#include <pthread.h>
#include <stdlib.h>
#include <assert.h>

struct arg_s {
  int i;      //indice de la case
  int* t_in;  //tableau des entrées
  int* t_out; //tableau des résulats
};
typedef struct arg_s arg;

void* moins_un(void* ptr_args) {
  arg* a = (arg*) ptr_args;
  a->t_out[a->i] = a->t_in[a->i] - 1;
  return NULL;
}

int main(){ //remplit res tq pour i\in[1..n], res[i] = val[i]-1
  int n = 10;
  int* val = (int*) malloc(n*sizeof(int));
  int* res = (int*) malloc(n*sizeof(int));

  //1-préparer les arguments
  arg* args = (arg*) malloc(n*sizeof(struct arg_s));
  for(int k = 0; k < n; k++){
    args[k] = (arg) {k,val,res};
  }
  //2-lancer les threads
  pthread_t* threads = (pthread_t*) malloc(n*sizeof(pthread_t));
  for(int k = 0; k < n; k++){
    pthread_create(threads+k, NULL, moins_un, args+k);
  }
  //3-attendre la fin des threads
  for(int k = 0; k < n; k++){
    pthread_join(threads[k], NULL);
  }
  //4-lire les résultats
  assert((res[0] == val[0]-1) && (res[n-1] == val[n-1]-1));
  //5-libérer la mémoire
  free (val);
  free (res);
  free (args);
  free (threads);
  return 0;
}
```

Cette notice fait suite à la notice "Lancer **deux** fils d'exécution en parallèle en C ".