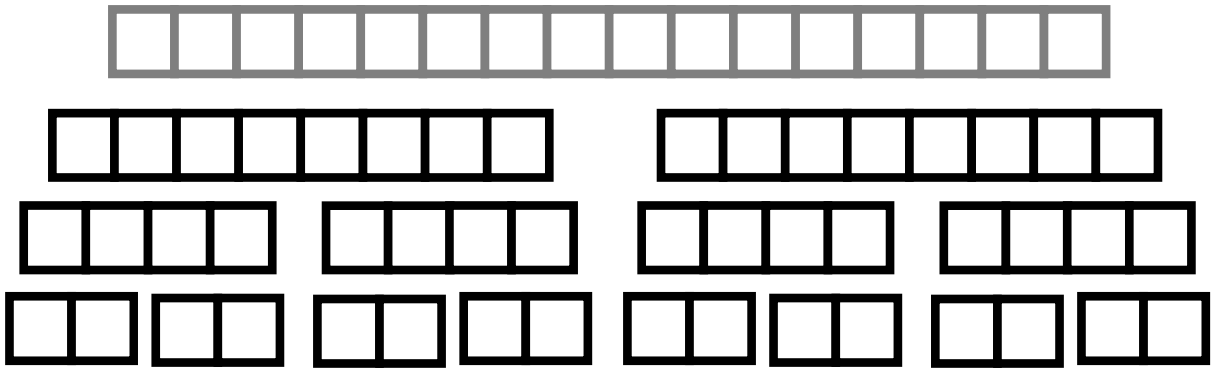
Tri Fusion : ordre de complexité

Éléments de l'algorithme à analyser pour une liste de taille **n** en entrée

// traitement des cas de terminaison	**Constant O(1)**
// divide	**Constant O(1)**
mid ← n/2	
// conquer	
Left ← **tri_fusion**(L(1 à mid), n/2)	**Quasi-linéaire**
Right ← **tri_fusion**(L(mid+1 à n), n/2+n%2)	**Linéarithmique** **O(nlog(n))**
// fusion des 2 sous-listes	
Sortir : **fusion_listes** (Left, n/2, Right, n/2+n%2)	**Linéaire O(n)**

Analyse de l'étape «conquer» :

Profondeur	**Entrée: taille n**
1	**2^1 x n/ 2^1 → n**
2	**2^2 x n/ 2^2 → n**
3	**2^3 x n/ 2^3 → n**



A chaque profondeur, le coût calcul est la somme des coût de fusion des sous-listes: leur nombre 2 fois plus grand est compensé par leur taille 2 fois plus petite. Bilan: O(n) par niveau de profondeur.

Combien de niveau de profondeurs faut-il compter ?

-> de l'ordre de $\log_2(n)$ pour atteindre le critère de terminaison. Bilan: $O(n\log_2(n))$