NODE SIZE

Parent offset int = 4 bytes

Metadata:
# keys in node
Int = 4 bytes
boolean: leaf?
bool = 1 byte

Child offset int = 4 bytes

Btree Obj
key
long = 8bytes
frequency
Int = 4bytes

Child offset int = 4 bytes

Btree Obj
key
long = 8bytes
frequency
Int = 4bytes

Child offset int = 4 bytes

Btree Obj
key
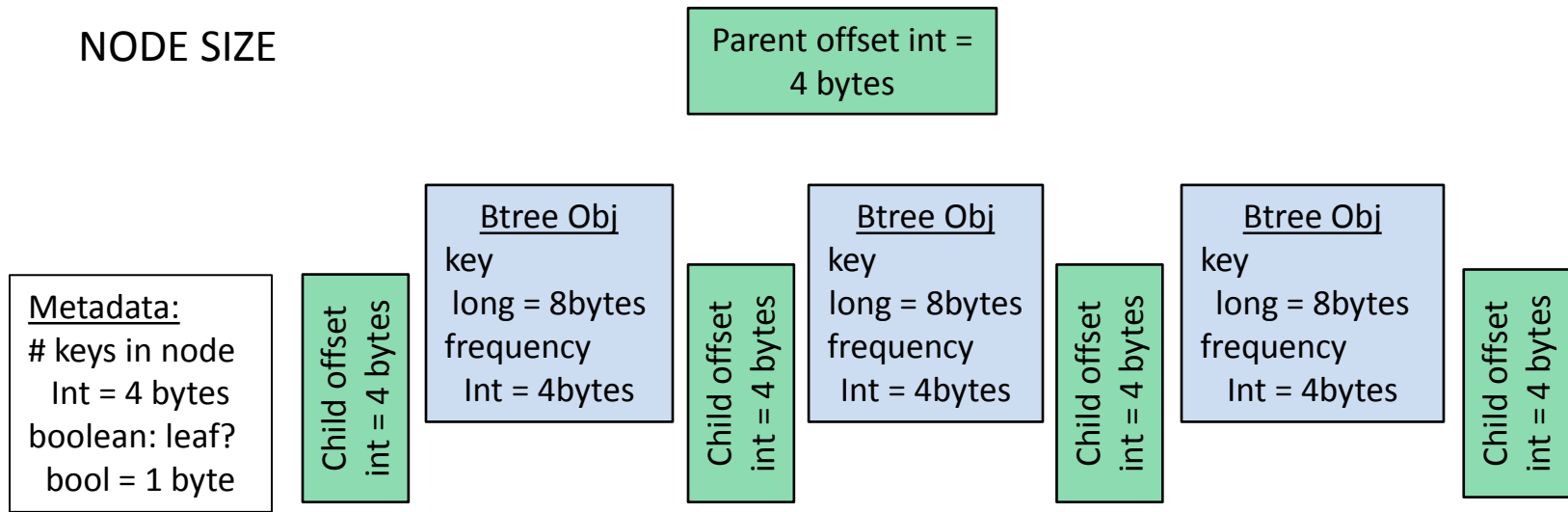long = 8bytes
frequency
Int = 4bytes

Child offset int = 4 bytes

Total storage space required depends on variable t (degree)

$(2t - 1) (12 \text{ bytes})$ = $24t - 12$ Btree objects/keys
$(2t + 1) (4 \text{ bytes})$ = $8t + 4$ children and parent pointers
(5 bytes) = 5 metadata

=$32t - 3$ bytes per node

Unless we want to add another 3 bytes in metadata to bring it to an even 32t bytes.

Now we have to decide on layout:
Metadata first, then parent, child, obj, child, obj?
Metadata first then parent, then all children, then all obj?
Other?