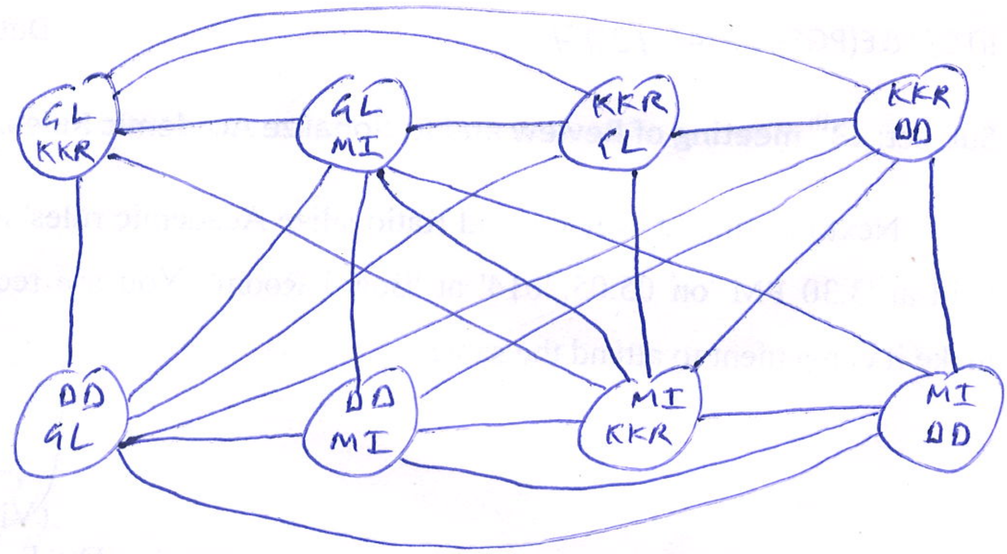
1. Create a graph with nodes corresponding to games to be played, and links to games that cannot co-occur:

(Home team mentioned above)



Now this is a graph colouring problem: Each colour will give us a set of games that can be held within a single 2-day window.

Let us colour the above graph greedily, as discussed in class: (starting top-left)

Colour 1 - { GL KKR, DD MI }

Colour 2 - { GL MI, KKR DD }

Colour 3 - { KKR GL, MI DD }

Colour 4 - { DD GL, MI KKR }

Hence we will need <u>4</u> (non-overlapping) 2-day windows.

(In this example it is easy to see that the greedy solution is optimal. There cannot be more than 2 games in any window, as 2 games involve all 4 teams.)

Within each (non-overlapping) window, games can always be kept <u>on the first day</u>, to avoid any overlap between successive windows.

Hence the final schedule is: (home first)

colour 1 →
Day 1 : GL vs. KKR, DD vs. MI
Day 2 : —

colour 2 →
Day 3 : GL vs. MI, KKR vs. DD
Day 4 : —

colour 3 →
Day 5 : KKR vs. GL, MI vs. DD
Day 6 : —

colour 4 →
Day 7 : DD vs. GL, MI vs. KKR

Finally, we need <u>7 days</u> to complete the tournament.

```
2. LIST merge (LIST A, LIST B) {

    LIST C = new LIST;

    int aptr = FIRST(A);
    int bptr = FIRST(B);
    int cptr = FIRST(C);

    while (aptr != END(A) and bptr != END(B))
    {
        int a = RETRIEVE(aptr, A);
        int b = RETRIEVE(bptr, B);
        if (a ≤ b){
            INSERT(a, cptr, C);
            aptr = NEXT(aptr, A);
            cptr = NEXT(cptr, C);
        }
        else {
            INSERT(b, cptr, C);
            bptr = NEXT(bptr, B);
            cptr = NEXT(cptr, C);
        }
    }
    while (aptr != END(A)) {
        INSERT(RETRIEVE(aptr, A), cptr, C);
        aptr = NEXT(aptr, A);
        cptr = NEXT(cptr, C);
    }
```

```
    while ( bptr != END(B)) {
        INSERT(RETRIEVE(bptr, B), cptr, C);
        bptr = NEXT (bptr, B);
        cptr = NEXT (cptr, C);
    }
    return C;
}
```

3. (a) Suppose $T(n)$ is $\Omega(n \log n)$, i.e., we can take $T(k) \geq a k \log k + b$ [set $b = T(1)$] for $k < n$, and try induction:

$$T(n) = T(n/3) + T(2n/3) + cn$$

$$\geq a\frac{n}{3}\log\frac{n}{3} + b + a\frac{2n}{3}\log\frac{2n}{3} + b + cn$$

$$\geq \frac{an}{3}(\log n - \log 3) + b + \frac{2an}{3}(\log n + \log 2 - \log 3) + b + cn$$

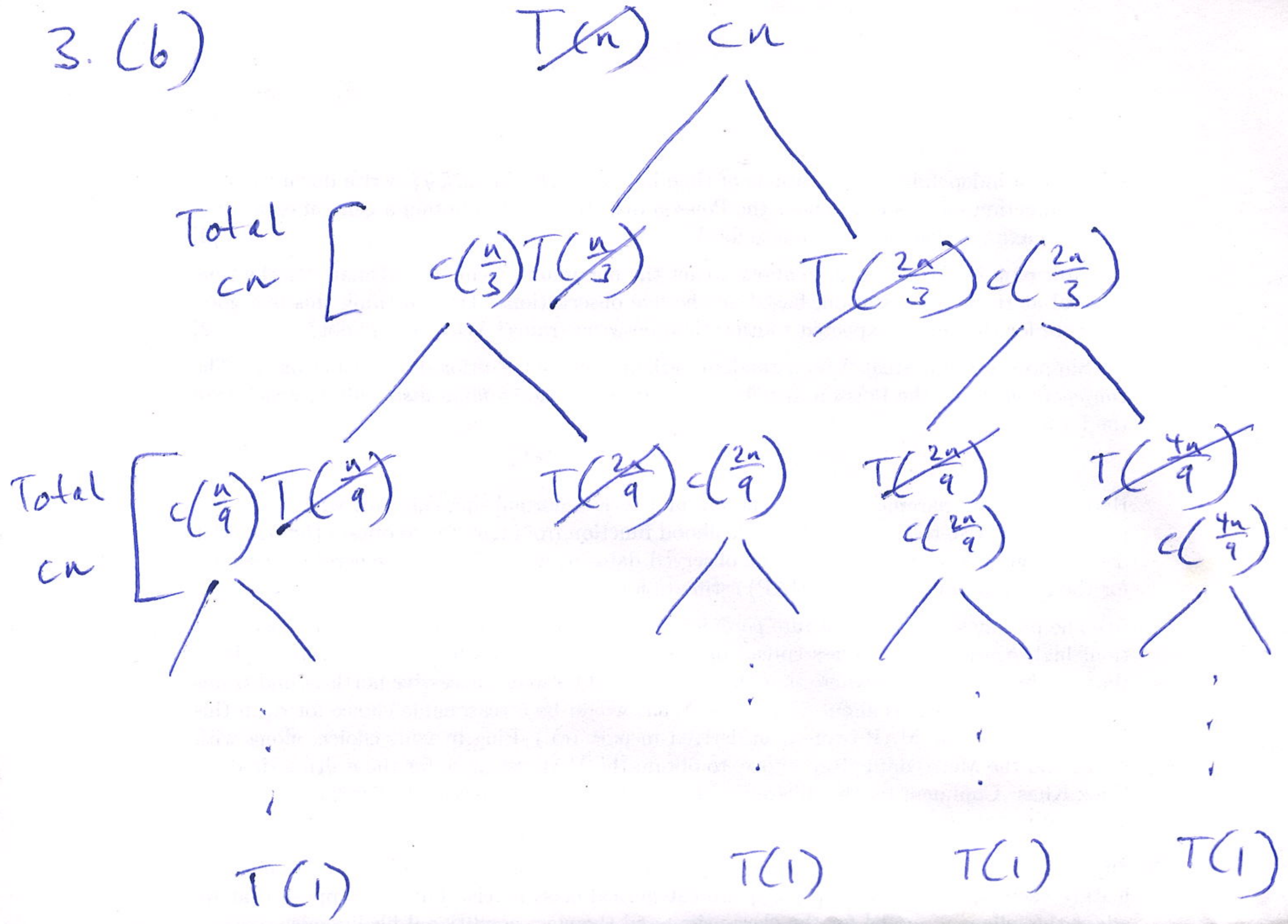
$$\geq an\log n - an\log 3 + \frac{2an}{3}\log 2 + 2b + cn$$

So we need:

$\left(-a\log 3 + \frac{2a}{3}\log 2 + c\right)$ to be non-negative. Choose

$$a\left(\frac{2}{3}\log 2 - \log 3\right) + c \geq 0 \Rightarrow a \leq \frac{c}{\left(\log 3 - \frac{2}{3}\log 2\right)}$$

3. (b)



Depth of tree: we need $\left(\frac{2}{3}\right)^i n = 1$, where i is the depth, so:

$$i = \log_{2/3} \frac{1}{n} = \log_{3/2} n$$

And each level has cost cn.

Hence total cost is $\Omega(n \log n)$.

(Even for shortest branch, depth is $\log_3 n$. Hence at least $\log_3 n$ levels with total cost cn per level.)