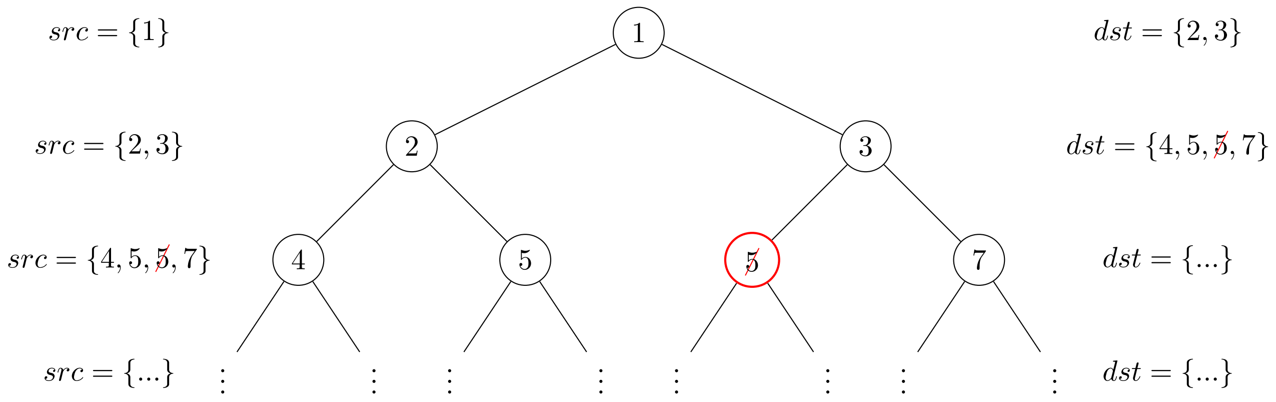


Listing 1: selectRecursive

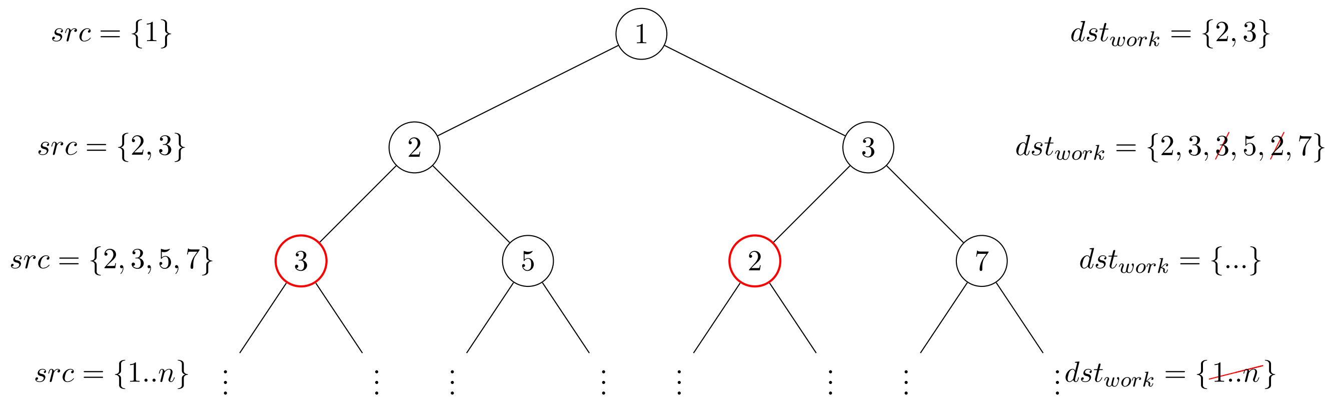
```
SELECT DISTINCT(dst) FROM team22.relation_facebook WHERE src IN(
    SELECT DISTINCT(dst) FROM team22.relation_facebook WHERE src IN(
        SELECT DISTINCT(dst) FROM team22.relation_facebook WHERE src IN(1)
    )
)
```

Listing 2: selectWithJoin

```
SELECT DISTINCT(rf3.dst)
    FROM public.relation_facebook rf1,
        public.relation_facebook rf2,
        public.relation_facebook rf3
    WHERE rf2.src = rf1.dst
        AND rf3.src = rf2.dst
        AND rf1.src = 765;
```

Listing 3: StoredProcedure

```
CREATE OR REPLACE FUNCTION recursivesearch(tInput integer[], iRecursionDepth integer, sTable text) RETURNS SETOF
    integer AS $$
    Declare
    intermDst_ integer[];
    iCount integer;
    BEGIN
    CREATE TABLE intermDst AS SELECT * FROM unnest(tInput);
    EXECUTE 'CREATE TABLE intermDst1 AS SELECT DISTINCT(dst) FROM ' || sTable || ' WHERE src IN (SELECT * FROM
        intermDst)';
    -- Does not return from function!
    return query SELECT * FROM intermDst1;
    -- Does not return from function!
    intermDst_ := ARRAY(SELECT * FROM intermDst1);
    raise notice 'timestamp: %', clock_timestamp();
    SELECT count(*) INTO iCount FROM intermDst;
    raise notice 'Count Table: %', iCount;
    DROP TABLE intermDst;
    DROP TABLE intermDst1;
    if iRecursionDepth > 1 THEN
    return query SELECT * FROM recursivesearch(intermDst_, iRecursionDepth - 1, sTable);
    ELSE
    RETURN;
    END IF;
    END;
$$ LANGUAGE plpgsql;
```



Listing 4: selectWithUnion

```
WITH RECURSIVE graphtraverse AS(
    SELECT DISTINCT(dst)
        FROM
            public.relation_facebook
        WHERE
            src =765
    UNION
    SELECT p.dst
        FROM
            relation_facebook p
        WHERE
            src IN ( p.src )
    )
SELECT * FROM graphtraverse
```