- Closure of a set of attributes X with respect to F is the set X^+ of all attributes that are functionally determined by X.
  i.e. $X^+ = \{ A \mid X \rightarrow A \text{ is in } F^+ \}$

- X^+ can be calculated by repeatedly applying A1, A2, A3 using the FDs in F

OR by using the following algorithm:

Algorithm 14.1 Determining X^+

```
X+ := X;
Repeat
  oldX+ := X+
  for each FD Y -> Z in F do
    if X+ is a superset of Y then
       X+ := X+ U Z;
Until (X+ = oldX+)
```

Example: F = { SSN -> ENAME,
PNUMBER-> {PNAME,PLOCATION},
{SSN,PNUMBER} -> HOURS }

{SSN}+ = {SSN, ENAME}

{PNUMBER}+ = {PNUMBER,PNAME,PLOCATION}

{SSN,PNUMBER}+ =
{SSN,PNUMBER,ENAME,PNAME,PLOCATION,HOURS}

2.3 Equivalence of Sets of FDs

- Two sets of FDs F and G are **equivalent** if:
  - every FD in F can be inferred from G, *and*
  - every FD in G can be inferred from F

- Hence, F and G are equivalent if $F^+=G^+$

- Definition: F **covers** G if every FD in G can be inferred from F (i.e., if G^+ subset-of F^+).

- F and G are equivalent if F covers G and G covers F

- There is an algorithm for checking equivalence of sets of FDs (which uses X+ algorithm).

2.4 Minimal Sets of FDs

- A set of FDs is **minimal** if it satisfies the following conditions:
  (1) Every dependency in F has a single attribute for its RHS.
  (2) We cannot remove any dependency from F and have a set of dependencies that is equivalent to F.
  (3) We cannot replace any dependency X -> A in F with a dependency Y -> A, where Y proper-subset-of X
  and still have a set of dependencies that is equivalent to F.

- Every set of FDs has an equivalent minimal set

- There can be several equivalent minimal sets

Algorithm 14.2 Finding minimal cover G for F

1. Set G := F
2. Replace each FD, X -> {A1,…,An} in G by n FDs
   X->A1, …, X->An
3. for each FD X->A in G,
   for each attribute B in X
     if (G – {X->A}) U {X-{B} -> A} is equivalent to G
     then replace X->A with (X-{B}->A) in G
3. For each remaining FD X->A in G
   If (G – {X->A}) is equivalent to G then remove X->A from G.

3.4 Third Normal Form

Definition:

- A relation schema R, F is in **third normal form** (**3NF**) if for each FD X->A in F+,
  (a) X is a superkey or
  (b) A belongs to a candidate key of R,F

BCNF (Boyce-Codd Normal Form)

- A relation schema R is in **Boyce-Codd Normal Form** (**BCNF**) if whenever an FD X -> A holds in R, then X is a superkey of R

- There exist relations that are in 3NF but not in BCNF

- The goal is to have each relation in BCNF (or 3NF)