Searching for Solutions

Careful Analysis of Expansions

The Bucket Algorithm

Solutions and Expansions

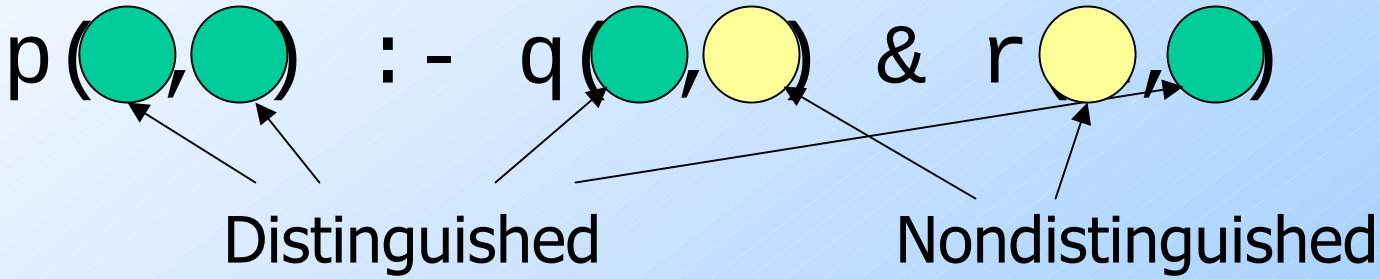
- For every subgoal p(X,Y) of a query, each solution must have a subgoal (view) whose expansion allows a containment mapping to p(X,Y).
- "Bucket" for a subgoal = set of views that "cover" the subgoal.
- A solution must include $\geq$ 1 view from each bucket.

But …

- There's much more to the story.
- A careful examination of how variables from the view definitions, query, solution, and expansion relate will eventually reveal additional constraints on the structure of the solutions.

(Non)Distinguished Variables

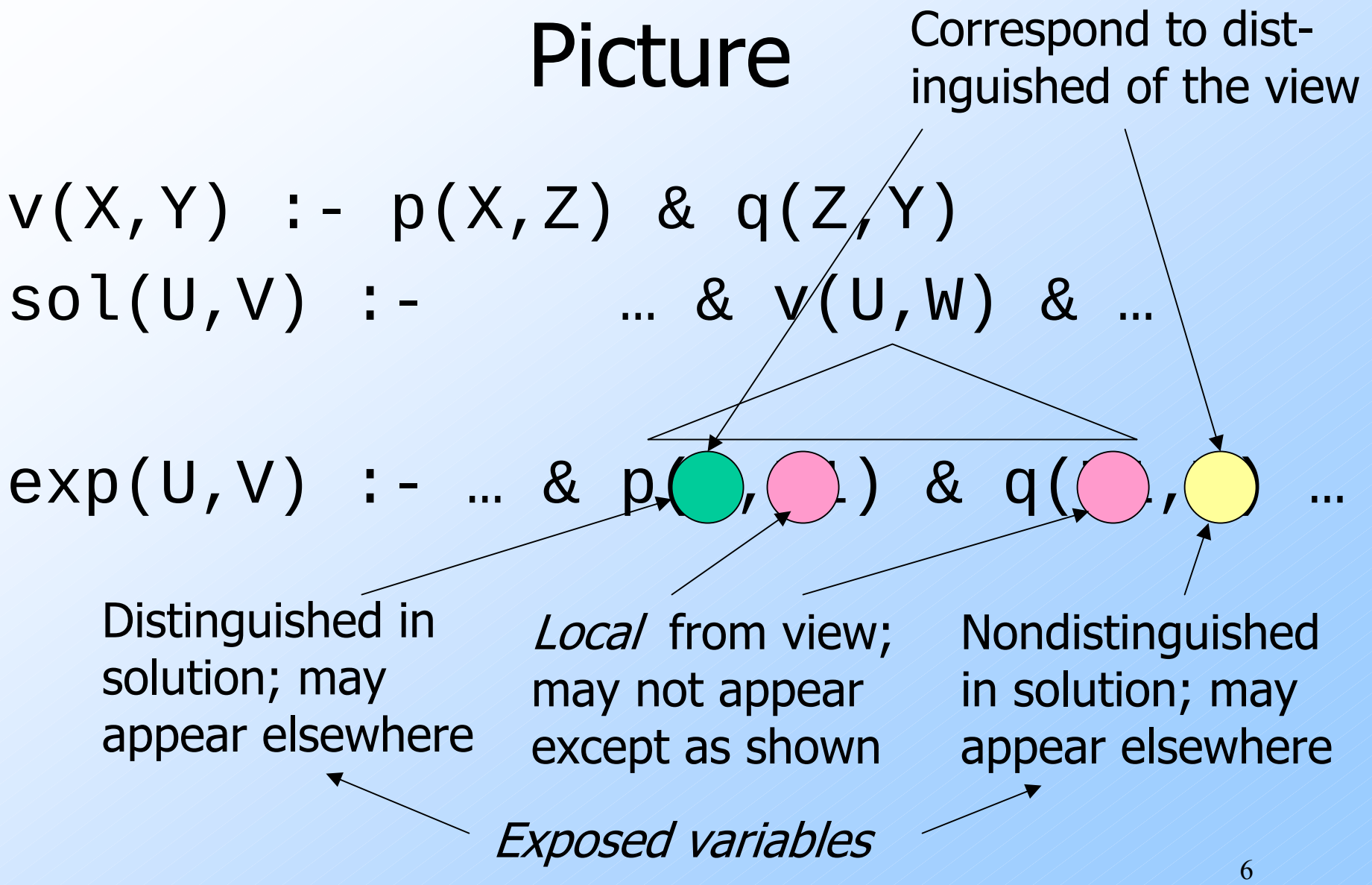
- A variable that appears in the head of a CQ is said to be *distinguished* ; otherwise *nondistinguished.*



Local Variables of Expansions

- When we expand a view subgoal of a solution, the nondistinguished variables of the view definition become *local*.
  - A local variable may not appear anywhere else in the expansion.
- Variables of the solution substitute for the distinguished variables of a view definition.

Picture


v(X,Y) :- p(X,Z) & q(Z,Y)
sol(U,V) :- … & v(U,W) & …
exp(U,V) :- … & p(,) & q(,) …
Correspond to dist-inguished of the view
Distinguished in solution; may appear elsewhere
Local from view; may not appear except as shown
Nondistinguished in solution; may appear elsewhere
Exposed variables

Exposed Variables

- Variables of the expansion that have substituted for distinguished variables of a view.
- These are the only variables that may appear in subgoals belonging to the expansion of two different solution subgoals.

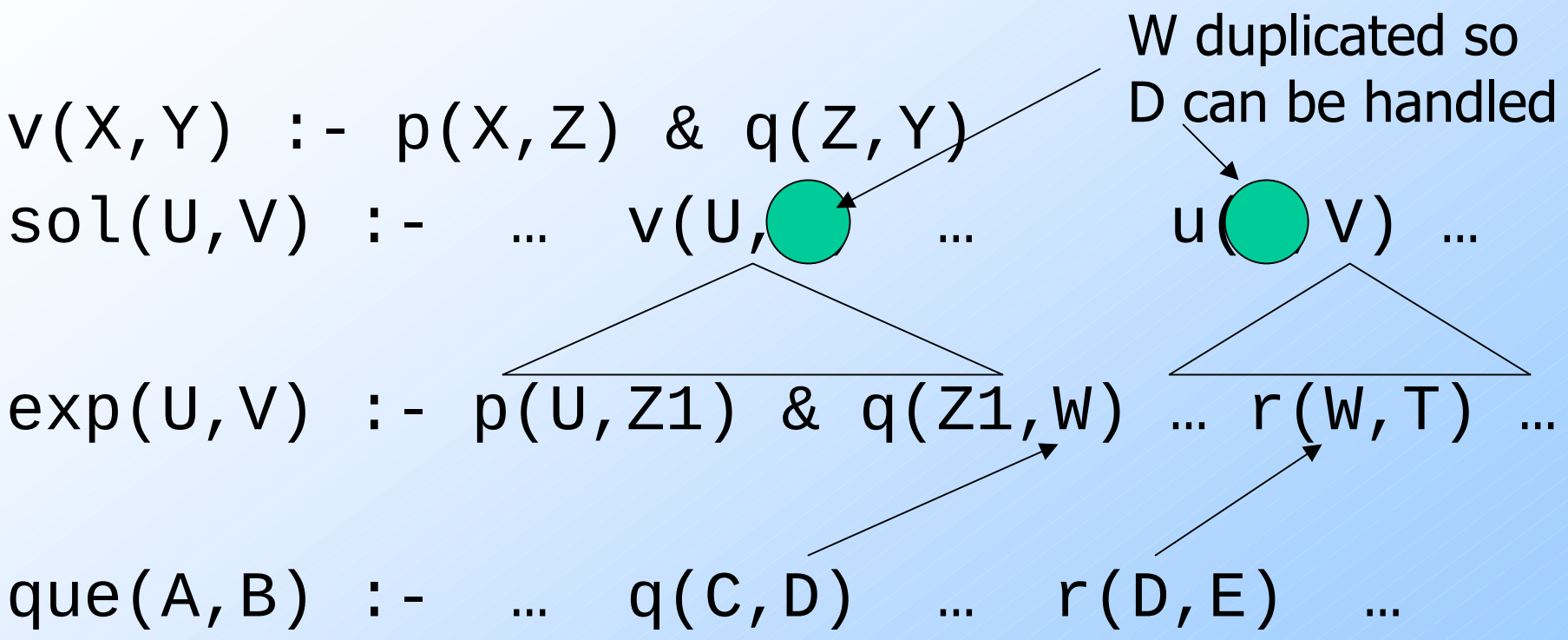
The Variables of the Query

- A query variable is *shared* if it appears more than once; otherwise it is *unique*.
- A distinguished query variable can only map to the corresponding distinguished variable of the expansion/solution.
- A nondistinguished, unique variable of the query maps to any variable of the expansion.

Mapping Shared Variables

- There are two options for shared variables:
  1. Map to a local variable of one expansion.
  2. Map to an exposed variable.
- Only in case (2) can the query subgoals with a shared variable map to expansion subgoals that come from more than one solution subgoal.

Picture --- Map to Exposed



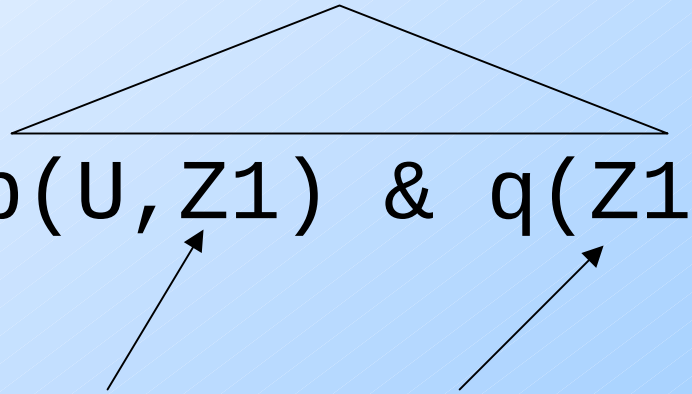
Shared variable D maps to exposed variable W. We can map another occurrence of D to a copy of W that comes from another view.

Picture

```
v(X,Y) :- p(X,Z) & q(Z,Y)
sol(U,V) :-      … & v(U,W) & …

exp(U,V) :- … & p(U,Z1) & q(Z1,W) …

que(U,V) :- … p(U,A) & q(A,D) …
```



All occurrences of shared variable
A map to local variable Z1.

Buckets

- To help search for solutions, we create buckets:
  1. One bucket for each subgoal of the query.
  2. One bucket for each shared variable in the query.

Buckets for Subgoals

- Members of the bucket for a subgoal p(A,B) are pairs consisting of:
  1. A view *v*.
  2. A particular *p* -subgoal in the body of *v*.
- There are conditions on p(A,B) and the target subgoal p(X,Y) described on the next slide.

Buckets for Subgoals --- (2)

1. p(A,B) must be mappable to p(X,Y). That is, if A=B, then X=Y.
2. If (say) A is a distinguished variable of the query, then X is distinguished in the view.
3. If (say) A is a shared variable, then X is distinguished in the view.

- Obvious extension to $\geq$ 2 arguments.

Buckets for Shared Variables

- Members of the bucket for a shared variable *A* consist of:
  1. A view *v*, and
  2. A set of subgoals *S* of *v* such that there is a CM from all the query subgoals containing *A* to *S*.
- In this mapping, distinguished variables of the query map to distinguished variables of the view.

Example

```
v(X,Y) :- p(X,Z) & p(Z,Y)
w(U,V) :- p(U,S) & p(S,T) & p(T,V)
q(A,B) :- p(A,C) & p(C,D) & p(D,E)
          & p(E,F) & p(F,G) & p(G,B)
```

- v = “grandparent”; w = “great-grandparent”; query q = “sixth-generation ancestors.”

Example --- p(A,C)

```
v(X,Y) :- p(X,Z) & p(Z,Y)
w(U,V) :- p(U,S) & p(S,T) & p(T,V)
q(A,B) :- p(A,C) & p(C,D) & p(D,E)
          & p(E,F) & p(F,G) & p(G,B)
```

- The bucket for p(A,C) is empty.
  - *A* is distinguished; *C* is shared.
  - No view subgoal has distinguished variables in both positions.

Example --- p(C,D)

```
v(X,Y) :- p(X,Z) & p(Z,Y)
w(U,V) :- p(U,S) & p(S,T) & p(T,V)
q(A,B) :- p(A,C) & p(C,D) & p(D,E)
          & p(E,F) & p(F,G) & p(G,B)
```

- The bucket for p(C,D) is empty.
  - Both *C* and *D* are shared.
  - No view subgoal has distinguished variables in both positions.
- Likewise, all subgoals of *q* have empty buckets.

Example --- Shared Variable *C*

```
v(X,Y) :- p(X,Z) & p(Z,Y)
w(U,V) :- p(U,S) & p(S,T) & p(T,V)
q(A,B) :- p(A,C) & p(C,D) & p(D,E)
          & p(E,F) & p(F,G) & p(G,B)
```

- The bucket for *C*:
  1. {p(X,Z), p(Z,Y)} from *v*.
     - Important: *X* is distinguished (since *A* maps to *X*).
  2. {p(U,S), p(S,T)} from *w*.
     - Important: *U* is distinguished (since *A* maps to *U*).

Shared Variable *C* --- Continued

```
v(X,Y) :- p(X,Z) & p(Z,Y)
w(U,V) :- p(U,S) & p(S,T) & p(T,V)
q(A,B) :- p(A,C) & p(C,D) & p(D,E)
          & p(E,F) & p(F,G) & p(G,B)
```

- The bucket for *C* does not contain {p(S,T), p(T,V)} from *w*.
  - Because distinguished variable *A* of the query would have to map to *S*, which is local in the view definition.

Example --- Shared Variable *D*

```
v(X,Y) :- p(X,Z) & p(Z,Y)
w(U,V) :- p(U,S) & p(S,T) & p(T,V)
q(A,B) :- p(A,C) & p(C,D) & p(D,E)
          & p(E,F) & p(F,G) & p(G,B)
```

- The bucket for *D*:
  1. {p(X,Z), p(Z,Y)} from *v*.
  2. {p(U,S), p(S,T)} and {p(S,T), p(T,V)} from *w*.
     - Either is OK, since neither *C* nor *E* is distinguished.
- *E*, *F* like *D*; *G* like *A*.

Example --- Continued

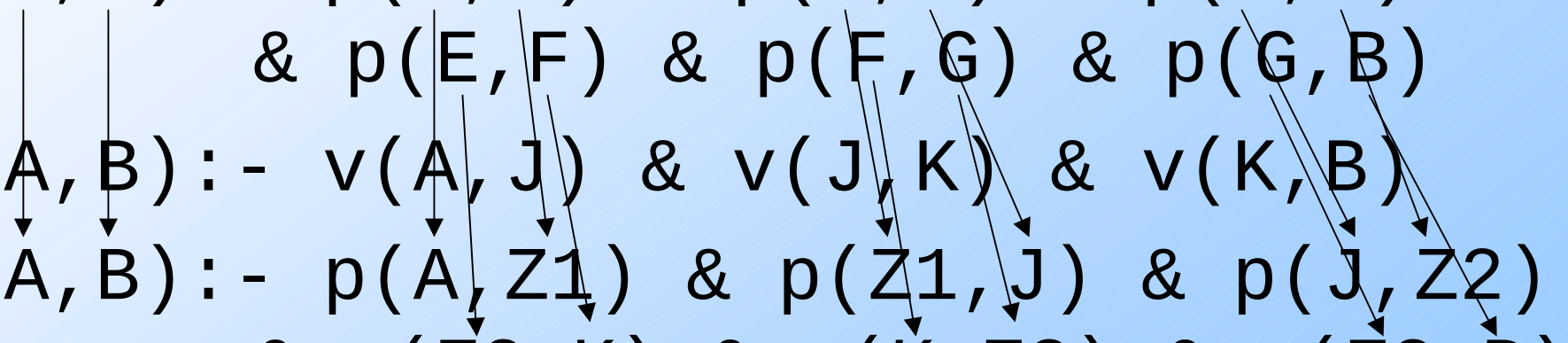
- Each of the six query subgoals must be covered by at least one member of a bucket.
- Since the subgoals themselves have empty buckets, we must group them according to their shared variables and cover them, in groups, from the buckets for the variables.

Example --- Continued

- One possibility: use the members from *v* in the buckets for *C*, *E*, and *G*.
- Since shared variables D and F map to distinguished variables of the view definition, we can use *v* three times in the solution, and equate the corresponding variables.

First Solution

```
v(X,Y):- p(X,Z) & p(Z,Y)
w(U,V):- p(U,S) & p(S,T) & p(T,V)
q(A,B):- p(A,C) & p(C,D) & p(D,E)
       & p(E,F) & p(F,G) & p(G,B)
s(A,B):- v(A,J) & v(J,K) & v(K,B)
e(A,B):- p(A,Z1) & p(Z1,J) & p(J,Z2)
       & p(Z2,K) & p(K,Z3) & p(Z3,B)
```



Example --- Continued

- Another possibility is to use one copy of w to cover the first three query subgoals and another copy of w to cover the last three.
- The first copy covers shared variables *C* and *D*; the second covers *F* and *G*.
- Shared variable *E* maps to distinguished variables of w.

Second Solution

```
v(X,Y) :- p(X,Z) & p(Z,Y)
w(U,V) :- p(U,S) & p(S,T) & p(T,V)
q(A,B) :- p(A,C) & p(C,D) & p(D,E)&
              p(E,F) & p(F,G) & p(G,B)
s(A,B) :- w(A,J) & w(J,B)
e(A,B) :- p(A,S1) & p(S1,T1) & p(T1,J) &
              p(J,S2) & p(S2,T2) & p(T2,B)
```

Why There Are No More Solutions

- For instance, we cannot use one *v* subgoal v(A,J) in the solution to cover shared variable *C* and another v(K,L) to cover *D*.
- v(A,J) expands to p(A,Z1) & p(Z1,J), forcing *D* to map to *J*.
- But v(K,L) expands to p(K,Z2) & p(Z2,L), forcing *D* to map to Z2.