Algorithm 1: Construct1

```
Input: EventStream $S$, Attribute Ranges $rs$, Predicates $preds$
Output: Graph $graph$
1  $EV \leftarrow \varnothing$;
2  $ARVS \leftarrow mkArray(len(rs))$ ;                    // the set of attribute ranges
3  $FES \leftarrow mkArray(len(rs))$ ;                     // the set of out-going edges
4  $TES \leftarrow mkArray(len(rs))$ ;                     // the set of in-going ranges
5  for $i \leftarrow 0$ until $len(rs)$ do
6      if $preds[i].operator \neq$ "eq" then
7          $ARVS[i] \leftarrow rs[i]$
8      else
9          $ARVS[i] \leftarrow \varnothing$
10     $FES[i] \leftarrow \varnothing$ ;
11     $TES[i] \leftarrow \varnothing$ ;
12 for $e$ in $S$ do
13     $EV \leftarrow EV \cup e$ for $i \leftarrow 0$ until $len(rs)$ do
14         $lval \leftarrow getLeftOprand(e)$;
15         $rval \leftarrow getRightOprand(e)$;
16         $FES[i] \leftarrow FES[i] \cup \{lval \rightarrow e\}$;
17         $AVS[i] \leftarrow AVS[i] \cup lval$;
18         if $preds[i].operator$ = "eq" then
               // equal is easy, use no dynamic range
19             $ARVS[i] \leftarrow ARVS[i] \cup rval$;
20             $TES[i] \leftarrow TES[i] \cup \{e \rightarrow rval\}$;
21         else
               // not equal, cut range into two and clone edges
22             $r \leftarrow$ findMatchRange($ARVS[i]$, $rval$);
23             $ARVS[i] \leftarrow ARVS[i] - r$;
24             $r1, r2 \leftarrow split(r, preds[i].operator)$ ;               // split range
25             $edges \leftarrow edgesEndWith(r)$;
26             $TES[i] \leftarrow TES[i] - edges$ ;
27             for $\{e' \rightarrow r'\}$ in $edges$ do
28                 $TES[i] \leftarrow TES[i] \cup \{e' \rightarrow r1\}$;
29                 $TES[i] \leftarrow TES[i] \cup \{e' \rightarrow r2\}$;
30             $ranges \leftarrow findRanges(ARVS[i], rval, preds[i][i].operator)$;
31             for $r'$ in $ranges$ do
32                 $TES[i] \leftarrow TES[i] \cup \{e \rightarrow r'\}$
33 $V \leftarrow EV$;
34 $E \leftarrow \varnothing$;
35 for $i \leftarrow 0$ until $len(FES)$ do
36     $V \leftarrow V \cup ARVS[i]$;
37     $E' = TES[i]$;
38     for $\{v \rightarrow e\}$ in $FES[i]$ do
39         $r \leftarrow inRange(ARVS[i])$;
40         $E' \leftarrow E' \cup \{r \rightarrow e\}$;
41     $E \leftarrow E \cup E'$;
42 $graph \leftarrow \{V, E\}$;
43 return $graph$;
```

Algorithm 2: Construct2

```
Input: EventStream $S$, Attribute Ranges $rs$, Predicates $preds$
Output: Graph $graph$
1  $EV \leftarrow \varnothing$;
2  $FES \leftarrow mkArray(len(rs))$ ;                    // the set of out-going edges
3  $TES \leftarrow mkArray(len(rs))$ ;                    // the set of in-going ranges
4  for $i \leftarrow 0$ until $len(rs)$ do
5      $FES[i] \leftarrow \varnothing$ ;
6      $TES[i] \leftarrow \varnothing$ ;
7  for $e$ in $S$ do
8      $EV \leftarrow EV \cup e$;
9      for $i \leftarrow 0$ until $len(rs)$ do
10         $lval \leftarrow getLeftOprand(e)$;
11         $rval \leftarrow getRightOprand(e)$;
12         $FES[i] \leftarrow FES[i] \cup \{lval \rightarrow e\}$;
           // non equal operators act as equal
13         $sval \leftarrow realVal(preds[i].operator, rval)$;
14         $TES[i] \leftarrow TES[i] \cup \{e \rightarrow sval\}$;
15 $V \leftarrow EV$;
16 $E \leftarrow \varnothing$;
17 for $i \leftarrow 0$ until $len(rs)$ do
       // split range based on the in-going edges
18     $KeyedTES \leftarrow keyedDest(TES[i])$;
19     $SortedKTES \leftarrow sortedByKey(keyedTES)$;
20     $gap \leftarrow 0$;
21     $ARV \leftarrow \varnothing$;
22     for $key$ in $SortedKTES$ do
23         $ARV \leftarrow ARV \cup [gap, key)$;
24         $gap \leftarrow key$;
       // reduce edges to ranges
25     $E' \leftarrow \varnothing$;
26     for $\{v \rightarrow e\}$ in $FES[i]$ do
27         $r \leftarrow inRange(ARV)$;
28         $E' \leftarrow E' \cup \{r \rightarrow e\}$;
29     for $\{e \rightarrow v\}$ in $TES[i]$ do
30         $rs' \leftarrow inRanges(ARV, preds[i].operator)$;
31         for $r$ in $rs'$ do
32             $E' \leftarrow E' \cup \{e \rightarrow r\}$;
33     $V \leftarrow V \cup ARV$;
34     $E \leftarrow E \cup E'$;
35 $graph \leftarrow \{V, E\}$;
36 return $graph$;
```