**Algorithm 3** Get corresponding ASG steps

**Input** ASG (V, E, S) /\* an attack scenario graph with vertex, edges, steps in   
 order \*/

SentList[] /\* a list of sentence segments for a CTI document \*/

**Output** MatchedAsgStepList[] /\* a list of ASG steps that metioned by the input CTI document \*/

SentOpList[] /\* a list represents verb-object pair

(SentOp) for CTI document \*/

AsgStepForDocsOpMap{} /\* a map stores which ASG steps are

corresponded to a SentOp \*/

/\* Generate regexes from ASG, denoted by RegSet \*/

Initialize *RegSet* []← get\_regex(*ASG*)

Initialize struct {

action, object, sentence, step\_number

} *OperationPair*

/\* Pick out sentences that contains any generated regexes \*/

*SentOpList* ← []

**for** *segment* in *SentList* **do**

**if** *segment* contains *reg\_i RegSet* **then**

**add** *OperationPair(*

*action* ← *None,*

*object* ← *reg\_i,*

*sentence* ← *segment)*

to *SentOpList*

**end if**

**end for**

/\* Extract the descriptive verb for every searched regex \*/

**for** *pair* in *SentOpList* **do**

*pair*.*action* ← get\_verb\_by\_dept\_parse(*pair*.*sentence*, *pair*.*object*)

**end for**

/\* Get mentioned ASG steps for the CTI document \*/

*MatchedAsgStepList* ← empty set()

*AsgStepForDocsOpMap* ← empty map()

**for** *op* in *SentOpList* **do**

**for** *step* in get\_steps(*ASG*) **do**

*correspond\_syscall\_list* ← query\_synonym\_base(*op.action*,

*op.object*) // see algo no.2

**if** *step.object* == *op.object* **&&** // verb & syscall matched

*step.action* *correspond\_syscall\_list* **then**

**add** *step* to *MatchedAsgStepList*

**add** key-value-pair {*op* -> *step.step\_number*} to

*AsgStepForDocsOpMap* // stores the mapping

**end if**

**end for**

**end for**

**return** *MatchedAsgStepList*.to\_list() // sort by step\_number

*SentOpList*

*AsgStepForDocsOpMap*