

1. Copyright.

Copyright © Dave Bone 1998 - 2015

2. *prt_xrefs_docs.lex* Grammar.

Various cross references:

- 1) used vocabulary symbols referenced against each rule's subrule and position
- 2) first set per rule
- 3) productions derived lr states
- 4) list of reducing lr states
- 5) Per state Follow set calculations
- 6) Commonized follow sets for reducing subrules

3. Fsm *Cprt_xrefs_docs* class.**4. *Cprt_xrefs_docs* op directive.**

```

< Cprt_xrefs_docs op directive 4 > ≡
  prt_sr_elems_filter_.insert(T_Enum :: T_T_subrule_def_);
  prt_sr_elems_filter_.insert(T_Enum :: T_referred_T_);
  prt_sr_elems_filter_.insert(T_Enum :: T_T_eosubrule_);
  prt_sr_elems_filter_.insert(T_Enum :: T_referred_rule_);
  prt_sr_elems_filter_.insert(T_Enum :: T_T_called_thread_eosubrule_);
  prt_sr_elems_filter_.insert(T_Enum :: T_T_null_call_thread_eosubrule_);
  no_subrules_per_rule_.push_back(0);
  time_t theTime = time(0);
  char *cTime = ctime(&theTime);
  gened_date_time_ += string(cTime);
  int n = gened_date_time_.find(' \n ');
  gened_date_time_[n] = '␣';
  rule_def_ = 0;
  subrule_def_ = 0;
  rule_no_ = 0;
  subrule_no_ = 0;
  elem_no_ = 0;
  no_of_rules_ = 0;
  no_of_subrules_ = 0;
  w_index_filename_ += grammar_filename_prefix_.c_str();
  w_index_filename_ += "_idx.w";
  ow_index_file_.open(w_index_filename_.c_str(), ios_base :: out | ios :: trunc);
  if (¬ow_index_file_) {
    CAbs_lr1_sym * sym = new Err_bad_filename(w_index_filename_.c_str());
    sym→set_who_created(__FILE__, __LINE__);
    parser_→add_token_to_error_queue(*sym);
    parser_→set_stop_parse(true);
    return;
  }

```

5. Cprt_xrefs_docs user-declaration directive.

```

⟨ Cprt_xrefs_docs user-declaration directive 5 ⟩ ≡
public: char big_buf[BIG_BUFFER_32K]; set < int > prt_sr_elems_filter_;
        std::map < int , std::string > xlated_names_; std::map < int ,
        std::list < state_element *>> productions_derived_states_list_;
std::vector < state *> reducing_states_list_; std::vector < int > no_subrules_per_rule_;

std::string ened_date_time_;
std::string w_index_filename_;
std::ofstream ow_index_file_;
std::string grammar_filename_prefix_;
std::string fq_filename_noext_;

int rule_no_;
int subrule_no_;
int elem_no_;
int no_of_rules_;
int no_of_subrules_;

std::string rule_name_;
std::string elem_name_;
rule_def * rule_def_;
T_subrule_def * subrule_def_;
std::list < NS_yacco2_terminals::rule_def *> rules_for_fs_prt_;
std::map < std::string , std::list < std::string >> xref_of_used_symbols_;

void add_symbol_to_xref_map(std::string & Key , std::string & Ref);
void determine_closure_derived_states();
void prt_states_follow_set();
void prt_follow_set_local_yield(follow_element * Fe);
void prt_follow_set_creators(follow_element * Fe);
void prt_follow_set_transitions(follow_element * Fe);
void prt_follow_set_merges(follow_element * Fe);
void prt_state_s_follow_set_rules(state * Cur_state);
void prt_common_follow_set_la();

```

6. Cprt_xrefs_docs user-implementation directive.

```

⟨Cprt_xrefs_docs user-implementation directive 6⟩ ≡
  void Cprt_xrefs_docs::determine_closure_derived_states(){ std::map < int
    , std::list < state_element *>> ::iterator xi; std::map < int ,
    std::list < state_element *>> ::iterator xie;
    STATES_ITER_type si = LR1_STATES.begin();
    STATES_ITER_type sie = LR1_STATES.end();
    int integerize_the_subrule(0);
    for (; si ≠ sie; ++si) { /* read states */
      state * cur_state = *si;
      using namespace NS_yacco2_T_enum;
      S_VECTOR_ITER_type svi = cur_state->state_s_vector_.begin();
      S_VECTOR_ITER_type svie = cur_state->state_s_vector_.end();
      S_VECTOR_ITER_type tvi = cur_state->state_s_vector_.find(-T_Enum::T_T_eosubrule_);
      if (tvi ≠ svie) {
        reducing_states_list_.push_back(cur_state);
        goto rd_vector_s_elems;
      }
      tvi = cur_state->state_s_vector_.find(-T_Enum::T_T_called_thread_eosubrule_);
      if (tvi ≠ svie) {
        reducing_states_list_.push_back(cur_state);
        goto rd_vector_s_elems;
      }
      tvi = cur_state->state_s_vector_.find(-T_Enum::T_T_null_call_thread_eosubrule_);
      if (tvi ≠ svie) {
        reducing_states_list_.push_back(cur_state);
        goto rd_vector_s_elems;
      }
    }
    rd_vector_s_elems: ;
    for (; svi ≠ svie; ++svi) { /* rd the same vector's elements */
      S_VECTOR_ELEMS_ITER_type seli = svi->second.begin();
      S_VECTOR_ELEMS_ITER_type selie = svi->second.end();
      for (; seli ≠ selie; ++seli) {
        state_element * se = *seli;
        if (se->previous_state_ ≠ 0) continue;
        integerize_the_subrule = se->subrule_def->its_grammar_s_pos();
        xi = productions_derived_states_list_.find(integerize_the_subrule);
        if (xi ≡ productions_derived_states_list_.end()) {
          productions_derived_states_list_[integerize_the_subrule] = std::list < state_element *> ();
        }
        xi = productions_derived_states_list_.find(integerize_the_subrule);
        xi->second.push_back(se);
      }
    }
  }
}
}
}

```

7. *add_symbol_to_xref_map*.

⟨More code 7⟩ ≡

```

void Cprt_xrefs_docs::add_symbol_to_xref_map(std::string & Key, std::string & Ref)
{
    std::map < std::string, std::list < std::string >> ::iterator i;
    i = xref_of_used_symbols_.find(Key.c_str());
    if (i ≡ xref_of_used_symbols_.end()) {
        xref_of_used_symbols_[Key.c_str()] = std::list < std::string > ();
        i = xref_of_used_symbols_.find(Key);
        std::list < std::string > &xxx = i->second;
        xxx.push_back(string(Ref.c_str()));
        return;
    }
    else {
        std::list < std::string > &xxx = i->second;
        xxx.push_back(string(Ref.c_str()));
    }
}

```

See also sections 8, 9, 10, 11, 12, 13, and 14.

8. *prt_follow_set_local_yield*.

⟨More code 7⟩ +≡

```

void Cprt_xrefs_docs::prt_follow_set_local_yield(follow_element * Fe)
{
    KCHARP w_follset_local_yield = "\\FollSetreducinglocalyield_□%s";
    sprintf(big_buf_, w_follset_local_yield, "□");
    ow_index_file_ << big_buf_ << endl;
    KCHARP w_follset_t = "%s";
    char t_name[Max_cweb_item_size];
    FOLLOW_SETS_ITER_type fsi = Fe->follow_set_.begin();
    FOLLOW_SETS_ITER_type fsie = Fe->follow_set_.end();
    for (; fsi ≠ fsie; ) { /* those Tes */
        T_in_stbl *t = *fsi;
        t_name[0] = (char) 0;
        XLATE_SYMBOLS_FOR_cweave(t->t_def()->t_name()->c_str(), t_name);
        sprintf(big_buf_, w_follset_t, t_name);
        ow_index_file_ << big_buf_;
        ++fsi;
        if (fsi ≠ fsie) {
            ow_index_file_ << ", " << endl;
        }
        else {
            ow_index_file_ << "." << endl;
        }
    }
    ow_index_file_ << endl; /* end it with blank line */
}

```

9. *prt_follow_set_creators.*

⟨More code 7⟩ +=

```

void Cprt_xrefs_docs::prt_follow_set_creators(follow_element * Fe){ char rule_name[Max_cweb_item_size];
    KCHARP w_follset_start_str = "\\halign{\n" "\\span\\FollSettemplate\n" "\\FollSettitle";
    KCHARP w_follset_stateno_rule = "{%s\\rulenameno{%i}}&\n" /* rule+rule no */
    "{"; /* start of the contributors */
    KCHARP w_follset_creators = "\\FollSetcreators{%i}{%i}{%i}";
    sprintf(big_buf_, w_follset_start_str, "");
    ow_index_file_ << big_buf_ << endl; rule_def * rd = ( rule_def * ) AST::content(*Fe->rule_def_t_);
    rule_name[0] = (char) 0;
    XLATE_SYMBOLS_FOR_cweave(rd->rule_name()->c_str(), rule_name);
    sprintf(big_buf_, w_follset_stateno_rule, rule_name, rd->rule_no());
    ow_index_file_ << big_buf_ << endl;
    SR_ELEMENTS_type::iterator sri = Fe->sr_elements_.begin();
    SR_ELEMENTS_type::iterator srie = Fe->sr_elements_.end(); for (; sri ≠ srie; ++sri) {
        /* follow set contributors */
        AST * et = *sri; /* eos */
        AST * prvt = et->pr_; refered_rule * rr = ( refered_rule * ) AST::content(*prvt);
        T_subrule_def * srd = rr->its_subrule_def();
        rule_def * sr_d = srd->its_rule_def();
        sprintf(big_buf_, w_follset_creators, sr_d->rule_no(), srd->subrule_no_of_rule(), rr->element_pos());
        ow_index_file_ << big_buf_ << endl; } }

```

10. *prt_follow_set_merges.*

⟨More code 7⟩ +=

```

void Cprt_xrefs_docs::prt_follow_set_merges(follow_element * Fe)
{
    KCHARP w_overflow_close_and_new_blank_rule = "\\cr\n" "{}&\n" /* blank rule name */
    "{%s"; /* start of new merging list */
    KCHARP w_follset_merges = "\\FollSetmerges{%i}"; /* state where other follow set rule lies */
    MERGES_ITER_type mi = Fe->merges_.begin();
    MERGES_ITER_type mie = Fe->merges_.end();
    int overflow_limit(10);
    int merge_cnt(0);
    for (; mi ≠ mie; ++mi) { /* transitions */
        state * s = *mi;
        ++merge_cnt;
        if (merge_cnt > overflow_limit) {
            sprintf(big_buf_, w_overflow_close_and_new_blank_rule, "␣");
            ow_index_file_ << big_buf_ << endl;
            merge_cnt = 1;
        }
        sprintf(big_buf_, w_follset_merges, s->state_no_);
        ow_index_file_ << big_buf_ << endl;
    }
}

```

11. *prt_follow_set_transitions.*

⟨More code 7⟩ +≡

```

void Cprt_xrefs_docs::prt_follow_set_transitions(follow_element * Fe){
    KCHARP w_follset_transitions = "\\FollSettransition{%i}{%i}"; /* rt bnded
    */
    TRANSITIONS_ITER_type ti = Fe-transitions_.begin();
    TRANSITIONS_ITER_type tie = Fe-transitions_.end(); for (; ti ≠ tie; ++ti) {
        /* transitions */
        follow_element * tfe = *ti; rule_def * rd = ( rule_def *) AST::content(*tfe-rd-def_t_);
        sprintf(big_buf_, w_follset_transitions, tfe-its_state-→state_no_, rd-→rule_no());
        ow_index_file_ << big_buf_ << endl; } KCHARP w_follset_end_str = "}%s\\cr";
        sprintf(big_buf_, w_follset_end_str, " ");
        ow_index_file_ << big_buf_ << endl;
        KCHARP w_follset_end_rule = "}%s";
        sprintf(big_buf_, w_follset_end_rule, " ");
        ow_index_file_ << big_buf_ << endl; }

```

12. *prt_state_s_follow_set_rules.*

⟨More code 7⟩ +≡

```

void Cprt_xrefs_docs::prt_state_s_follow_set_rules(state * Cur_state)
{
    KCHARP w_follset_stateno = /* stateno */
    "\\FollSetstateno{%i}";
    S_FOLLOW_SETS_ITER_type fsi = Cur_state-→state_s_follow_set_map_.begin();
    S_FOLLOW_SETS_ITER_type fsie = Cur_state-→state_s_follow_set_map_.end();
    if (fsi ≡ fsie) return; /* nada follow set info in this state */
    sprintf(big_buf_, w_follset_stateno, Cur_state-→state_no_);
    ow_index_file_ << big_buf_ << endl;
    for (; fsi ≠ fsie; ++fsi) { /* state's follow set info */
        follow_element * fe = (*fsi).second;
        prt_follow_set_creators(fe);
        prt_follow_set_merges(fe);
        prt_follow_set_transitions(fe);
        prt_follow_set_local_yield(fe);
    }
}

```

13. *prt_states_follow_set.*

⟨More code 7⟩ +≡

```

void Cprt_xrefs_docs::prt_states_follow_set()
{
    KCHARP w_states_follow_sets = "@**_Lr1_State's_Follow_sets_and_reducing_lookahead_sets.\\f\\
    break\\n" "\\FollSetnotesintro\\n" "\\fbreak";
    ow_index_file_ << w_states_follow_sets << std::endl;
    STATES_ITER_type si = LR1_STATES.begin();
    STATES_ITER_type sie = LR1_STATES.end();
    for (; si ≠ sie; ++si) { /* walk the states */
        state * cur_state = *si;
        prt_state_s_follow_set_rules(cur_state);
    }
}

```

14. *prt_common_follow_set_la.*

⟨More code 7⟩ +≡

```

void Cprt_xrefs_docs::prt_common_follow_set_la()
{
    KCHARP w_common_follow_sets = "@*1CommonFollowsets.\\fbreak\n";
    ow_index_file_ << w_common_follow_sets << std::endl;
    KCHARP w_common_follow_set = "@*2LAset:␣%i.\\fbreak\n" "\\item{}\\n" "\\raggedright";
    KCHARP la_set_entry_literal = "%s";
    char t_name[Max_cweb_item_size];
    COMMON_LA_SETS_ITER_type i = COMMON_LA_SETS.begin();
    COMMON_LA_SETS_ITER_type ie = COMMON_LA_SETS.end();
    for (int idx = 0; i ≠ ie; ++i, ++idx) {
        LA_SET_type * la_set = *i;
        sprintf(big_buf_, w_common_follow_set, idx + 1);
        ow_index_file_ << big_buf_ << endl;
        LA_SET_ITER_type j = la_set->begin(); /* list out the T literals */
        LA_SET_ITER_type je = la_set->end();
        for (; j ≠ je; ) {
            T_in_stbl * tsym = *j;
            t_name[0] = (char) 0;
            XLATE_SYMBOLS_FOR_cweave(tsym->t_def()->t_name()->c_str(), t_name);
            sprintf(big_buf_, la_set->entry_literal, t_name);
            ow_index_file_ << big_buf_;
            ++j;
            if (j ≠ je) {
                ow_index_file_ << ", " << endl;
            }
            else {
                ow_index_file_ << "." << endl;
            }
        }
        ow_index_file_ << endl; /* close off the items */
    }
}

```

15. *Cprt_xrefs_docs user-prefix-declaration directive.*

⟨Cprt_xrefs_docs user-prefix-declaration directive 15⟩ ≡

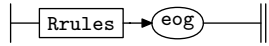
```

#include "time.h"
#include "o2_externs.h"
#include "prt_sr_elements.h"
extern void PRT_RULE_S_FIRST_SET(std::ofstream & Ofile, NS_yacco2_terminals::rule_def * Rule_def);
extern void XLATE_SYMBOLS_FOR_cweave(const char *Sym_to_xlate, char *Xlated_sym);
extern int MPOST_CWEB_xlated_symbol(AST * Sym, char *Xlated_sym);
extern STATES_type LR1_STATES;
extern COMMON_LA_SETS_type COMMON_LA_SETS;

```


16. *Rprt_xrefs_docs* rule.

Rprt_xrefs_docs



(Rprt_xrefs_docs subrule 1 op directive 16) ≡

```

Cprt_xrefs_docs * fsm = ( Cprt_xrefs_docs * ) rule_info_.parser_--fsm_tbl_;
int rule_no = 1;
KCHARP w_doc_index = "\\input_\"supp-pdf\"\\n\"\\input_\"/usr/local/yacco2/diagrams+et\
c/o2mac.tex\"\\n\"\\IDXdoctitle{%s}{%s}\";
char xlate_file[Max_cweb_item_size];
XLATE_SYMBOLS_FOR_cweave(fsm-w_index_filename_.c_str(), xlate_file);
sprintf(fsm-big_buf_, w_doc_index, xlate_file, xlate_file);
fsm-ow_index_file_ << fsm-big_buf_ << std::endl;
KCHARP w_used_xref_index = "@*_Grammar_symbols:Used_cross_referen\
ce.\\fbreak\n\"Reference_of_each_grammar's_symbol_used_within_each_rule's_\
productions.The_index\n\"uses_the_tripple:rule_name,its_subrul\
e_no,and_the_symbol's_position\n\"within_the_symbol_string.\";
fsm-ow_index_file_ << w_used_xref_index << std::endl;
char key[Max_cweb_item_size];
char xlate_key_sym[Max_cweb_item_size];
char xref_key[Max_cweb_item_size];
char xlate_sym[Max_cweb_item_size];
char xlated_rule[Max_cweb_item_size];
std::map < std::string, std::list < std::string >> ::iterator i = fsm-xref_of_used_symbols_.begin();
std::map < std::string, std::list < std::string >> ::iterator ie = fsm-xref_of_used_symbols_.end();
for ( ; i ≠ ie; ++i ) {
    strcpy(key, i-first.c_str());
    XLATE_SYMBOLS_FOR_cweave((const char *) key, xlate_key_sym);
    fsm-ow_index_file_ << "@*2_\" << xlate_key_sym << \":.\\fbreak\" << endl;
    std::list < std::string > &xxx = i-second;
    std::list < std::string > ::iterator k = xxx.begin();
    std::list < std::string > ::iterator ke = xxx.end();
    for ( ; k ≠ ke; ++k ) {
        strcpy(xref_key, k-c_str());
        XLATE_SYMBOLS_FOR_cweave((const char *) xref_key, xlate_sym);
        fsm-ow_index_file_ << xlate_sym << "\\_\" << endl;
    }
    fsm-ow_index_file_ << "\\fbreak\" << endl;
}
KCHARP w_fs_index = "@*_Grammar_Rules's_First_Sets.\\fbreak\n";
fsm-ow_index_file_ << w_fs_index << std::endl;
std::list < rule_def * > ::iterator j = fsm-rules_for_fs_prt_.begin();
std::list < rule_def * > ::iterator je = fsm-rules_for_fs_prt_.end();
for ( ; j ≠ je; ++j ) {
    PRT_RULE_S_FIRST_SET(fsm-ow_index_file_, *j);
}
;
KCHARP w_lr_state_network = "@*2_LR_State_Network.\\fbreak\n\"\\LRstaternetwork\n";
KCHARP w_xref_rule_rank_to_literal = "@.R$_{i}$_---_s@>\n";
fsm-ow_index_file_ << w_lr_state_network << std::endl;

```

```

fsm-determine_closure_derived_states(); std::map < int , std::list < state_element *>> ::iterator xi;
    std::map < int , std::list < state_element *>> ::iterator xie;
xi = fsm-productions_derived_states_list_.begin();
xie = fsm-productions_derived_states_list_.end();
rule_def * ord(0);
rule_def * rd(0); for ( ; xi ≠ xie; ++xi) { /* walk all the derived productions list */
std::list < state_element * > &selist = xi-second;
std::list < state_element * > ::iterator f1st_el = selist.begin(); state_element * se = ( state_element * )
    *f1st_el;
T_subrule_def * srd = se-subrule_def_;
rd = srd-its_rule_def();
if (ord ≠ rd) {
    ord = rd;
    XLATE_SYMBOLS_FOR_cweave(ord-rule_name()-c_str(), xlated_rule);
    KCHARP w_rule_name = "@*3_□s.\\fbreak";
    sprintf(fsm-big_buf_, w_rule_name, xlated_rule);
    fsm-ow_index_file_ << fsm-big_buf_ << std::endl;
    sprintf(fsm-big_buf_, w_xref_rule_rank_to_literal, rd-rule_no(), xlated_rule);
    fsm-ow_index_file_ << fsm-big_buf_ << std::endl;
}
KCHARP w_subrule = "\\Subrulestartsymstrindent{%i}□";
sprintf(fsm-big_buf_, w_subrule, srd-subrule_no_of_rule());
fsm-ow_index_file_ << fsm-big_buf_; /* print its rhs elements */
AST * sr_t = srd-subrule_s_tree();
tok_can_ast_functor sr_elems_walk_functr;
ast_prefix_1forest prt_sr_elems_walk(*sr_t, &sr_elems_walk_functr, &fsm-prt_sr_elems_filter_,
    ACCEPT_FILTER);
tok_can < AST * > prt_sr_elems_can(prt_sr_elems_walk);
using namespace NS_prt_sr_elements;
Cprt_sr_elements prt_sr_elements_fsm;
prt_sr_elements_fsm.ow_index_file_ = &fsm-ow_index_file_;
Parser prt_sr_elements(prt_sr_elements_fsm, &prt_sr_elems_can, 0);
prt_sr_elements.parse();
list < state_element * > &dlist = xi-second;
list < state_element * > ::iterator yi = dlist.begin();
list < state_element * > ::iterator yie = dlist.end();
KCHARP w_subrule_derived_states = "\\Subrulederivedstatesindent□";
KCHARP w_merged = "{\\Mergedstate{%i}}"; std::set < int > chk_merge;
chk_merge.clear(); for ( ; yi ≠ yie; ++yi) { /* derive state list per closed production */
fsm-ow_index_file_ << w_subrule_derived_states << endl;
state_element * se = *yi;
state_element * dse = se; for ( ; dse ≠ 0; dse = dse-next_state_element_) {
    /* walk the derived plank */
fsm-ow_index_file_ << dse-self_state_-state_no_; if (dse-next_state_element_ ≠ 0) { std::set <
    int > ::iterator si = chk_merge.find(dse-next_state_element_-self_state_-state_no_);
if (si ≠ chk_merge.end()) {
    sprintf(fsm-big_buf_, w_merged, dse-goto_state_-state_no_);
    fsm-ow_index_file_ << fsm-big_buf_ << endl;
    break;
}
} chk_merge.insert(dse-self_state_-state_no_);

```

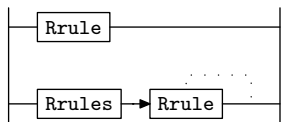
```

fsm-ow_index_file_ << "\\_\\_" << endl; } } KCHARP w_list_of_reduced_states =
    "@*2_List_of_reducing_states.\\fbreak\\n" "\\Listofreducingstates\\n" "\\fbreak\\n";
fsm-ow_index_file_ << w_list_of_reduced_states << std::endl;
fsm-ow_index_file_ << "\\Reducedstatelist" << std::endl;
std::vector < state * > ::iterator ri = fsm-reducing_states_list_.begin();
std::vector < state * > ::iterator rie = fsm-reducing_states_list_.end();
for ( ; ri != rie; ++ri ) {
    state * s = *ri;
    fsm-ow_index_file_ << '{' << s-state_no_;
    switch (s-state_type_) {
    case 0:
        { /*shift only */
            break;
        }
    case 1:
        {
            fsm-ow_index_file_ << "\\Reduceonly";
            break;
        }
    case 2:
        {
            fsm-ow_index_file_ << "\\ShiftReduce";
            break;
        }
    case 3:
        {
            fsm-ow_index_file_ << "\\MultipleReduces";
            break;
        }
    case 4:
        {
            fsm-ow_index_file_ << "\\ShiftandMultipleReduces";
            break;
        }
    }
    fsm-ow_index_file_ << "\\_\\_" << std::endl;
}
fsm-prt_states_follow_set();
fsm-prt_common_follow_set_la();
fsm-ow_index_file_ << "@*_Index." << endl;
fsm-ow_index_file_.close();

```

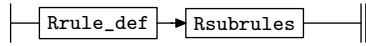
17. Rrules rule.

Rrules

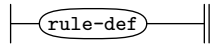


18. *Rrule* rule.

Rrule

**19. *Rrule_def* rule.**

Rrule_def



⟨ Rrule_def subrule 1 op directive 19 ⟩ ≡

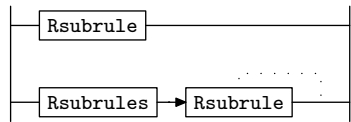
```

Cprt_xrefs_docs * fsm = ( Cprt_xrefs_docs * ) rule_info_.parser_-fsm_tbl_;
fsm-rule_def_ = sf-p1--;
fsm-rules_for_fs_prt_.push_back(fsm-rule_def_);
++fsm-rule_no_;
fsm-subrule_no_ = 0;

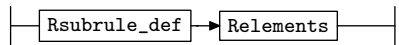
```

20. *Rsubrules* rule.

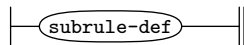
Rsubrules

**21. *Rsubrule* rule.**

Rsubrule

**22. *Rsubrule_def* rule.**

Rsubrule_def



⟨ Rsubrule_def subrule 1 op directive 22 ⟩ ≡

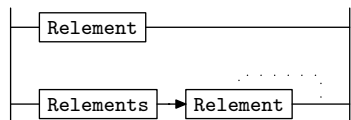
```

Cprt_xrefs_docs * fsm = ( Cprt_xrefs_docs * ) rule_info_.parser_-fsm_tbl_;
++fsm-subrule_no_;
fsm-elem_no_ = 0;
fsm-subrule_def_ = sf-p1--;

```

23. *Relements* rule.

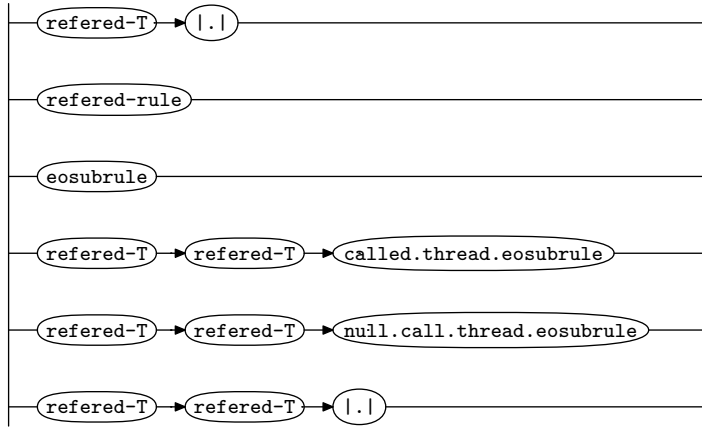
Relements



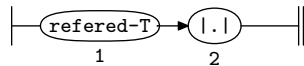
24. Relement rule.

Use of |.| to make grammar lr(1).

Relement



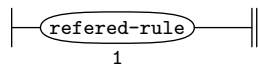
25. Relement's subrule 1.



< Relement subrule 1 op directive 25 > ≡

```
Cprt_xrefs_docs * fsm = ( Cprt_xrefs_docs * ) rule_info_.parser_--fsm_tbl_;
++fsm_elem_no_;
string xref_entry;
const char *xref_pattern = "%s□i.%i";
sprintf (fsm_big_buf_, xref_pattern, fsm_rule_def->rule_name()-c_str(), fsm_subrule_no_, fsm_elem_no_);
xref_entry += fsm_big_buf_;
string xref_key (sf-p1_--its_t_def()-t_name()-c_str());
fsm_add_symbol_to_xref_map(xref_key, xref_entry);
```

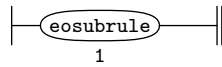
26. Relement's subrule 2.



< Relement subrule 2 op directive 26 > ≡

```
Cprt_xrefs_docs * fsm = ( Cprt_xrefs_docs * ) rule_info_.parser_--fsm_tbl_;
++fsm_elem_no_;
string xref_entry;
const char *xref_pattern = "%s□i.%i";
sprintf (fsm_big_buf_, xref_pattern, fsm_rule_def->rule_name()-c_str(), fsm_subrule_no_, fsm_elem_no_);
xref_entry += fsm_big_buf_;
string xref_key (sf-p1_--its_rule_def()-rule_name()-c_str());
fsm_add_symbol_to_xref_map(xref_key, xref_entry);
```

27. Relement's subrule 3.



⟨ Relement subrule 3 op directive 27 ⟩ ≡

```

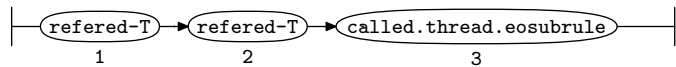
Cprt_xrefs_docs * fsm = ( Cprt_xrefs_docs * ) rule_info_.parser_--fsm_tbl_;
++fsm-elem_no_;
if ( fsm-elem_no_ ≡ 1 ) { /* epsilon */
    string xref_entry;

    const char * xref_pattern = "%s□%i.%i";

    sprintf( fsm-big_buf_, xref_pattern, fsm-rule_def->rule_name()-c_str(), fsm-subrule_no_, fsm-elem_no_ );
    xref_entry += fsm-big_buf_;
    string xref_key( "\\emptyrule" );
    fsm-add_symbol_to_xref_map( xref_key, xref_entry );
}

```

28. Relement's subrule 4.



⟨ Relement subrule 4 op directive 28 ⟩ ≡

```

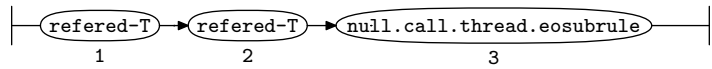
Cprt_xrefs_docs * fsm = ( Cprt_xrefs_docs * ) rule_info_.parser_--fsm_tbl_;
++fsm-elem_no_;
string xref_entry;

const char * xref_pattern = "%s□%i.%i";

sprintf( fsm-big_buf_, xref_pattern, fsm-rule_def->rule_name()-c_str(), fsm-subrule_no_, fsm-elem_no_ );
xref_entry += fsm-big_buf_;
string xref_key;
if ( sf-p1--its_t_def()-enum_id() ≡ T_Enum::T_LR1_parallel_operator_ ) {
    xref_key += "|||";
}
else {
    xref_key += "|t|";
}
fsm-add_symbol_to_xref_map( xref_key, xref_entry );
++fsm-elem_no_;
string xref_rtned_entry;
sprintf( fsm-big_buf_, xref_pattern, fsm-rule_def->rule_name()-c_str(), fsm-subrule_no_, fsm-elem_no_ );
xref_rtned_entry += fsm-big_buf_;
string xref_rtned_key( sf-p2--its_t_def()-t_name()-c_str() );
fsm-add_symbol_to_xref_map( xref_rtned_key, xref_rtned_entry );
++fsm-elem_no_;
string xref_thd_entry;
sprintf( fsm-big_buf_, xref_pattern, fsm-rule_def->rule_name()-c_str(), fsm-subrule_no_, fsm-elem_no_ );
xref_thd_entry += fsm-big_buf_;
string xref_thd_key;
xref_thd_key += sf-p3--ns()-identifier()-c_str();
xref_thd_key += " : ";
xref_thd_key += sf-p3--called_thread_name()-identifier()-c_str();
fsm-add_symbol_to_xref_map( xref_thd_key, xref_thd_entry );

```

29. Relement's subrule 5.



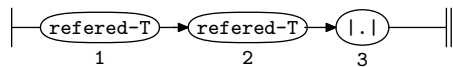
⟨ Relement subrule 5 op directive 29 ⟩ ≡

```

Cprt_xrefs_docs * fsm = ( Cprt_xrefs_docs * ) rule_info_.parser_--fsm_tbl_;
++fsm-elem_no_;
string xref_entry;
const char *xref_pattern = "%s□i.%i";
sprintf (fsm-big_buf_, xref_pattern, fsm-rule_def->rule_name()-c_str(), fsm-subrule_no_, fsm-elem_no_);
xref_entry += fsm-big_buf_;
string xref_key;
if (sf-p1--its_t_def()-enum_id() ≡ T_Enum::T_LR1_parallel_operator_) {
    xref_key += "|||";
}
else {
    xref_key += "|t|";
}
fsm-add_symbol_to_xref_map(xref_key, xref_entry);
++fsm-elem_no_;
string xref_rtnd_key(sf-p2--its_t_def()-t_name()-c_str());
string xref_rtnd_entry;
sprintf (fsm-big_buf_, xref_pattern, fsm-rule_def->rule_name()-c_str(), fsm-subrule_no_, fsm-elem_no_);
xref_rtnd_entry += fsm-big_buf_;
fsm-add_symbol_to_xref_map(xref_rtnd_key, xref_rtnd_entry);
++fsm-elem_no_;
string xref_thd_entry;
sprintf (fsm-big_buf_, xref_pattern, fsm-rule_def->rule_name()-c_str(), fsm-subrule_no_, fsm-elem_no_);
xref_thd_entry += fsm-big_buf_;
string xref_thd_key("NULL□thread");
fsm-add_symbol_to_xref_map(xref_thd_key, xref_thd_entry);

```

30. Relement's subrule 6.



⟨ Relement subrule 6 op directive 30 ⟩ ≡

```

Cprt_xrefs_docs * fsm = ( Cprt_xrefs_docs * ) rule_info_.parser_--fsm_tbl_;
++fsm-elem_no_;
string xref_entry;
const char *xref_pattern = "%s□i.%i";
sprintf (fsm-big_buf_, xref_pattern, fsm-rule_def->rule_name()-c_str(), fsm-subrule_no_, fsm-elem_no_);
xref_entry += fsm-big_buf_;
string xref_key(sf-p1--its_t_def()-t_name()-c_str());
fsm-add_symbol_to_xref_map(xref_key, xref_entry);
++fsm-elem_no_;
string xref_2_entry;
sprintf (fsm-big_buf_, xref_pattern, fsm-rule_def->rule_name()-c_str(), fsm-subrule_no_, fsm-elem_no_);
xref_2_entry += fsm-big_buf_;
string xref_2_key(sf-p2--its_t_def()-t_name()-c_str());
fsm-add_symbol_to_xref_map(xref_2_key, xref_2_entry);

```

31. First Set Language for O_2^{linker} .

```
/*
  File: prt_xrefs_docs.fsc
  Date and Time: Fri Jan  2 15:33:50 2015
*/
transitive      n
grammar-name    "prt_xrefs_docs"
name-space     "NS_prt_xrefs_docs"
thread-name    "Cprt_xrefs_docs"
monolithic     y
file-name      "prt_xrefs_docs.fsc"
no-of-T       569
list-of-native-first-set-terminals 1
  rule_def
end-list-of-native-first-set-terminals
list-of-transitive-threads 0
end-list-of-transitive-threads
list-of-used-threads 0
end-list-of-used-threads
fsm-comments
"Output xref doc --- \n'first set' per rule, and referenced symbols."
```


32. Lr1 State Network.

\Rightarrow				State: 1 state type: s		
\leftarrow	rule	\rightarrow	R# sr# Po	\leftarrow	subrule element	\rightarrow Brn Gto Red LA
c	Rrule_def		4 1 1		rule-def	1 2 2
c	Rprt_xrefs_docs		1 1 1		Rrules <u>eog</u>	1 3 4
c	Rrules		2 2 1		Rrules <u>Rrule</u>	1 3 5
c	Rrules		2 1 1		Rrule	1 23 23
c	Rrule		3 1 1		Rrule_def <u>Rsubrules</u>	1 6 8
\Rightarrow	<i>rule-def</i>			State: 2 state type: r		
\leftarrow	rule	\rightarrow	R# sr# Po	\leftarrow	subrule element	\rightarrow Brn Gto Red LA
t	Rrule_def		4 1 2			1 0 2 1
\Rightarrow	<i>Rrules</i>			State: 3 state type: s		
\leftarrow	rule	\rightarrow	R# sr# Po	\leftarrow	subrule element	\rightarrow Brn Gto Red LA
t	Rprt_xrefs_docs		1 1 2		eog	1 4 4
c	Rrule_def		4 1 1		rule-def	3 2 2
t	Rrules		2 2 2		Rrule	1 5 5
c	Rrule		3 1 1		Rrule_def <u>Rsubrules</u>	3 6 8
\Rightarrow	<i>eog</i>			State: 4 state type: r		
\leftarrow	rule	\rightarrow	R# sr# Po	\leftarrow	subrule element	\rightarrow Brn Gto Red LA
t	Rprt_xrefs_docs		1 1 3			1 0 4 2
\Rightarrow	<i>Rrule</i>			State: 5 state type: r		
\leftarrow	rule	\rightarrow	R# sr# Po	\leftarrow	subrule element	\rightarrow Brn Gto Red LA
t	Rrules		2 2 3			1 0 5 3
\Rightarrow	<i>Rrule_def</i>			State: 6 state type: s		
\leftarrow	rule	\rightarrow	R# sr# Po	\leftarrow	subrule element	\rightarrow Brn Gto Red LA
c	Rsubrule_def		7 1 1		subrule-def	6 7 7
t	Rrule		3 1 2		Rsubrules	3 8 8
c	Rsubrules		5 2 1		Rsubrules <u>Rsubrule</u>	6 8 9
c	Rsubrules		5 1 1		Rsubrule	6 22 22
c	Rsubrule		6 1 1		Rsubrule_def <u>Relements</u>	6 10 19
\Rightarrow	<i>subrule-def</i>			State: 7 state type: r		
\leftarrow	rule	\rightarrow	R# sr# Po	\leftarrow	subrule element	\rightarrow Brn Gto Red LA
t	Rsubrule_def		7 1 2			6 0 7 4
\Rightarrow	<i>Rsubrules</i>			State: 8 state type: s/r		
\leftarrow	rule	\rightarrow	R# sr# Po	\leftarrow	subrule element	\rightarrow Brn Gto Red LA
t	Rrule		3 1 3			3 0 8 3
c	Rsubrule_def		7 1 1		subrule-def	8 7 7
t	Rsubrules		5 2 2		Rsubrule	6 9 9
c	Rsubrule		6 1 1		Rsubrule_def <u>Relements</u>	8 10 19
\Rightarrow	<i>Rsubrule</i>			State: 9 state type: r		
\leftarrow	rule	\rightarrow	R# sr# Po	\leftarrow	subrule element	\rightarrow Brn Gto Red LA
t	Rsubrules		5 2 3			6 0 9 5

\Rightarrow <i>Rsubrule_def</i>		State: 10 state type: ^s	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA
c Relement	9 4 1	referred-T	10 11 15
c Relement	9 6 1	referred-T	10 11 14
c Relement	9 1 1	referred-T	10 11 12
c Relement	9 5 1	referred-T	10 11 16
c Relement	9 2 1	referred-rule	10 17 17
c Relement	9 3 1	eosubrule	10 18 18
t Rsubrule	6 1 2	Relements	8 19 19
c Relements	8 2 1	Relements <u>Relement</u>	10 19 20
c Relements	8 1 1	Relement	10 21 21
\Rightarrow <i>referred-T</i>		State: 11 state type: ^s	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA
t Relement	9 1 2	.	10 12 12
t Relement	9 4 2	referred-T	10 13 15
t Relement	9 6 2	referred-T	10 13 14
t Relement	9 5 2	referred-T	10 13 16
\Rightarrow <i> .</i>		State: 12 state type: ^r	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA
t Relement	9 1 3		10 0 12 6
\Rightarrow <i>referred-T</i>		State: 13 state type: ^s	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA
t Relement	9 6 3	.	10 14 14
t Relement	9 4 3	called thread eosubrule	10 15 15
t Relement	9 5 3	null call thread eosubrule	10 16 16
\Rightarrow <i> .</i>		State: 14 state type: ^r	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA
t Relement	9 6 4		10 0 14 6
\Rightarrow <i>calledthreadeosubrule</i>		State: 15 state type: ^r	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA
t Relement	9 4 4		10 0 15 6
\Rightarrow <i>nullcallthreadeosubrule</i>		State: 16 state type: ^r	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA
t Relement	9 5 4		10 0 16 6
\Rightarrow <i>referred-rule</i>		State: 17 state type: ^r	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA
t Relement	9 2 2		10 0 17 6
\Rightarrow <i>eosubrule</i>		State: 18 state type: ^r	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA
t Relement	9 3 2		10 0 18 6
\Rightarrow <i>Relements</i>		State: 19 state type: ^{s/r}	
← rule	→ R# sr# Po ←	subrule element	→ Brn Gto Red LA

t Rsubrule	6	1	3		8	0	19	5
c Relement	9	4	1	referred-T	19	11	15	
c Relement	9	6	1	referred-T	19	11	14	
c Relement	9	1	1	referred-T	19	11	12	
c Relement	9	5	1	referred-T	19	11	16	
c Relement	9	2	1	referred-rule	19	17	17	
c Relement	9	3	1	eosubrule	19	18	18	
t Relements	8	2	2	Relement	10	20	20	

⇒ *Relement* State: 20 state type: *r*
 ← rule → R# sr# Po ← subrule element → Brn Gto Red LA
 t Relements 8 2 3 10 0 20 6

⇒ *Relement* State: 21 state type: *r*
 ← rule → R# sr# Po ← subrule element → Brn Gto Red LA
 t Relements 8 1 2 10 0 21 6

⇒ *Rsubrule* State: 22 state type: *r*
 ← rule → R# sr# Po ← subrule element → Brn Gto Red LA
 t Rsubrules 5 1 2 6 0 22 5

⇒ *Rrule* State: 23 state type: *r*
 ← rule → R# sr# Po ← subrule element → Brn Gto Red LA
 t Rrules 2 1 2 1 0 23 3

33. Index.

|.|: 24.
 __FILE__: 4.
 __LINE__: 4.
 ACCEPT_FILTER: 16.
 add_symbol_to_xref_map: 5, 7, 25, 26, 27, 28, 29, 30.
 add_token_to_error_queue: 4.
 AST: 9, 11, 15, 16.
 ast_prefix_1forest: 16.
 begin: 6, 8, 9, 10, 11, 12, 13, 14, 16.
 big_buf: 5, 8, 9, 10, 11, 12, 14, 16, 25, 26, 27, 28, 29, 30.
 BIG_BUFFER_32K: 5.
 c_str: 4, 7, 8, 9, 14, 16, 25, 26, 27, 28, 29, 30.
 CAbs_lr1_sym: 4.
 called_thread_eosubrule: 24.
 called_thread_name: 28.
 chk_merge: 16.
 clear: 16.
 close: 16.
 COMMON_LA_SETS: 14, 15.
 COMMON_LA_SETS_ITER_type: 14.
 COMMON_LA_SETS_type: 15.
 content: 9, 11.
 Cpvt_sr_elements: 16.
 Cpvt_xrefs_docs: 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 19, 22, 25, 26, 27, 28, 29, 30.
 cTime: 4.
 ctime: 4.
 cur_state: 6, 13.
 Cur_state: 5, 12.
 determine_closure_derived_states: 5, 6, 16.
 dlist: 16.
 dse: 16.
 elem_name_: 5.
 elem_no_: 4, 5, 22, 25, 26, 27, 28, 29, 30.
 element_pos: 9.
 end: 6, 7, 8, 9, 10, 11, 12, 13, 14, 16.
 endl: 8, 9, 10, 11, 12, 13, 14, 16.
 enum_id: 28, 29.
 eog: 16.
 eosubrule: 24.
 Err_bad_filename: 4.
 et: 9.
 fe: 12.
 Fe: 5, 8, 9, 10, 11.
 find: 4, 6, 7, 16.
 first: 16.
 follow_element: 5, 8, 9, 10, 11, 12.
 follow_set: 8.
 FOLLOW_SETS_ITER_type: 8.
 fq_filename_noext: 5.
 fsi: 8, 12.
 fsie: 8, 12.
 fsm: 16, 19, 22, 25, 26, 27, 28, 29, 30.
 fsm_tbl_: 16, 19, 22, 25, 26, 27, 28, 29, 30.
 f1st_el: 16.
 gened_date_time_: 4, 5.
 goto_state_: 16.
 grammar_filename_prefix_: 4, 5.
 identifier: 28.
 idx: 14.
 ie: 14, 16.
 insert: 4, 16.
 integerize_the_subrule: 6.
 ios: 4.
 ios_base: 4.
 iterator: 6, 7, 9, 16.
 its_grammar_s_pos: 6.
 its_rule_def: 9, 16, 26.
 its_state_: 11.
 its_subrule_def: 9.
 its_t_def: 25, 28, 29, 30.
 je: 14, 16.
 KCHARP: 8, 9, 10, 11, 12, 13, 14, 16.
 ke: 16.
 Key: 5, 7.
 key: 16.
 la_set: 14.
 la_set_entry_literal: 14.
 LA_SET_ITER_type: 14.
 LA_SET_type: 14.
 lex: 2.
 list: 5, 6, 7, 16.
 LR1_STATES: 6, 13, 15.
 map: 5, 6, 7, 16.
 Max_cweb_item_size: 8, 9, 14, 16.
 merge_cnt: 10.
 merges_: 10.
 MERGES_ITER_type: 10.
 mi: 10.
 mie: 10.
 MPOST_CWEB_xlated_symbol: 15.
 n: 4.
 next_state_element_: 16.
 no_of_rules_: 4, 5.
 no_of_subrules_: 4, 5.
 no_subrules_per_rule_: 4, 5.
 ns: 28.
 NS_pvt_sr_elements: 16.
 NS_yacco2_T_enum: 6.
 NS_yacco2_terminals: 5, 15.

null call thread eosubrule: 24.
Ofile: 15.
ofstream: 5, 15.
open: 4.
ord: 16.
out: 4.
overflow_limit: 10.
ow_index_file_: 4, 5, 8, 9, 10, 11, 12, 13, 14, 16.
parse: 16.
Parser: 16.
parser_: 4, 16, 19, 22, 25, 26, 27, 28, 29, 30.
pr_: 9.
previous_state_: 6.
productions_derived_states_list_: 5, 6, 16.
prt_common_follow_set_la: 5, 14, 16.
prt_follow_set_creators: 5, 9, 12.
prt_follow_set_local_yield: 5, 8, 12.
prt_follow_set_merges: 5, 10, 12.
prt_follow_set_transitions: 5, 11, 12.
PRT_RULE_S_FIRST_SET: 15, 16.
prt_sr_elements: 16.
prt_sr_elements_fsm: 16.
prt_sr_elems_can: 16.
prt_sr_elems_filter_: 4, 5, 16.
prt_sr_elems_walk: 16.
prt_state_s_follow_set_rules: 5, 12, 13.
prt_states_follow_set: 5, 13, 16.
prt_xrefs_docs: 2.
push_back: 4, 6, 7, 19.
pvrt: 9.
p1_: 19, 22, 25, 26, 28, 29, 30.
p2_: 28, 29, 30.
p3_: 28.
rd: 9, 11, 16.
rd_vector_s_elems: 6.
reducing_states_list_: 5, 6, 16.
Ref: 5, 7.
referred-rule: 24.
referred-T: 24.
referred_rule: 9.
Relement: 23.
Relement: 24, 25, 26, 27, 28, 29, 30.
Relements: 23.
Relements: 21, 23.
ri: 16.
rie: 16.
Rprt_xrefs_docs: 16.
rr: 9.
Rrule: 18.
Rrule: 17.
Rrule_def: 18.
Rrule_def: 19.
Rrules: 16, 17.
Rrules: 17.
Rsubrule: 20.
Rsubrule: 21.
Rsubrule_def: 21.
Rsubrule_def: 22.
Rsubrules: 20.
Rsubrules: 18, 20.
rule-def: 19.
Rule_def: 15.
rule_def: 5, 9, 11, 15, 16.
rule_def_: 4, 5, 19, 25, 26, 27, 28, 29, 30.
rule_def_t_: 9, 11.
rule_info_: 16, 19, 22, 25, 26, 27, 28, 29, 30.
rule_name: 9, 16, 25, 26, 27, 28, 29, 30.
rule_name_: 5.
rule_no: 9, 11, 16.
rule_no_: 4, 5, 19.
rules_for_fs_prt_: 5, 16, 19.
S_FOLLOW_SETS_ITER_type: 12.
S_VECTOR_ELEMS_ITER_type: 6.
S_VECTORS_ITER_type: 6.
se: 6, 16.
second: 6, 7, 12, 16.
self_state_: 16.
seli: 6.
selie: 6.
selist: 16.
set: 5, 16.
set_stop_parse: 4.
set_who_created: 4.
sf: 19, 22, 25, 26, 28, 29, 30.
si: 6, 13, 16.
sie: 6, 13.
sprintf: 8, 9, 10, 11, 12, 14, 16, 25, 26, 27, 28, 29, 30.
sr_d: 9.
sr_elements_: 9.
SR_ELEMENTS_type: 9.
sr_elems_walk_funcptr: 16.
sr_t: 16.
srd: 9, 16.
sri: 9.
srie: 9.
state: 5, 6, 10, 12, 13, 16.
state_element: 5, 6, 16.
state_no_: 10, 11, 12, 16.
state_s_follow_set_map_: 12.
state_s_vector_: 6.
state_type_: 16.
STATES_ITER_type: 6, 13.
STATES_type: 15.

- std*: 5, 6, 7, 13, 14, 15, 16.
- strcpy*: 16.
- string*: 4, 5, 7, 16, 25, 26, 27, 28, 29, 30.
- subrule-def*: 22.
- subrule_def*: 4, 5, 6, 16, 22.
- subrule_no*: 4, 5, 19, 22, 25, 26, 27, 28, 29, 30.
- subrule_no_of_rule*: 9, 16.
- subrule_s_tree*: 16.
- svi*: 6.
- svie*: 6.
- Sym*: 15.
- sym*: 4.
- Sym_to_xlate*: 15.
- t_def*: 8, 14.
- T_Enum*: 4, 6, 28, 29.
- T_in_stbl*: 8, 14.
- T_LR1_parallel_operator*: 28, 29.
- t_name*: 8, 14, 25, 28, 29, 30.
- T_refered_rule*: 4.
- T_refered_T*: 4.
- T_subrule_def*: 5, 9, 16.
- T_T_called_thread_eosubrule*: 4, 6.
- T_T_eosubrule*: 4, 6.
- T_T_null_call_thread_eosubrule*: 4, 6.
- T_T_subrule_def*: 4.
- tfe*: 11.
- theTime*: 4.
- ti*: 11.
- tie*: 11.
- time*: 4.
- tok_can*: 16.
- tok_can_ast_functor*: 16.
- transitions*: 11.
- TRANSITIONS_ITER_type*: 11.
- true*: 4.
- trunc*: 4.
- tsym*: 14.
- tvi*: 6.
- vector*: 5, 16.
- w_common_follow_set*: 14.
- w_common_follow_sets*: 14.
- w_doc_index*: 16.
- w_follset_creators*: 9.
- w_follset_end_rule*: 11.
- w_follset_end_str*: 11.
- w_follset_local_yield*: 8.
- w_follset_merges*: 10.
- w_follset_start_str*: 9.
- w_follset_stateno*: 12.
- w_follset_stateno_rule*: 9.
- w_follset_t*: 8.
- w_follset_transitions*: 11.
- w_fs_index*: 16.
- w_index_filename*: 4, 5, 16.
- w_list_of_reduced_states*: 16.
- w_lr_state_network*: 16.
- w_merged*: 16.
- w_overflow_close_and_new_blank_rule*: 10.
- w_rule_name*: 16.
- w_states_follow_sets*: 13.
- w_subrule*: 16.
- w_subrule_derived_states*: 16.
- w_used_xref_index*: 16.
- w_xref_rule_rank_to_literal*: 16.
- xi*: 6, 16.
- xie*: 6, 16.
- xlate_file*: 16.
- xlate_key_sym*: 16.
- xlate_sym*: 16.
- XLATE_SYMBOLS_FOR_cweave*: 8, 9, 14, 15, 16.
- xlated_names*: 5.
- xlated_rule*: 16.
- Xlated_sym*: 15.
- xref_entry*: 25, 26, 27, 28, 29, 30.
- xref_key*: 16, 25, 26, 27, 28, 29, 30.
- xref_of_used_symbols*: 5, 7, 16.
- xref_pattern*: 25, 26, 27, 28, 29, 30.
- xref_rtned_entry*: 28, 29.
- xref_rtned_key*: 28, 29.
- xref_thd_entry*: 28, 29.
- xref_thd_key*: 28, 29.
- xref_2_entry*: 30.
- xref_2_key*: 30.
- xxx*: 7, 16.
- yi*: 16.
- yie*: 16.

< Cprt_xrefs_docs op directive 4 >
< Cprt_xrefs_docs user-declaration directive 5 >
< Cprt_xrefs_docs user-implementation directive 6 >
< Cprt_xrefs_docs user-prefix-declaration directive 15 >
< More code 7, 8, 9, 10, 11, 12, 13, 14 >
< Relement subrule 1 op directive 25 >
< Relement subrule 2 op directive 26 >
< Relement subrule 3 op directive 27 >
< Relement subrule 4 op directive 28 >
< Relement subrule 5 op directive 29 >
< Relement subrule 6 op directive 30 >
< Rprt_xrefs_docs subrule 1 op directive 16 >
< Rrule_def subrule 1 op directive 19 >
< Rsubrule_def subrule 1 op directive 22 >

prt_xrefs_docs Grammar

Date: January 2, 2015 at 15:38

File: prt_xrefs_docs.lex

Ns: NS_prt_xrefs_docs

Version: 1.0

Debug: false

Grammar Comments:

Type: Monolithic

Output xref doc — “first set” per rule, and referenced symbols.

	Section	Page
Copyright	1	1
<i>prt_xrefs_docs.lex</i> Grammar	2	2
Fsm Cprt_xrefs_docs class	3	2
Cprt_xrefs_docs op directive	4	2
Cprt_xrefs_docs user-declaration directive	5	3
Cprt_xrefs_docs user-implementation directive	6	4
<i>add_symbol_to_xref_map</i>	7	5
<i>prt_follow_set_local_yield</i>	8	5
<i>prt_follow_set_creators</i>	9	6
<i>prt_follow_set_merges</i>	10	6
<i>prt_follow_set_transitions</i>	11	7
<i>prt_state_s_follow_set_rules</i>	12	7
<i>prt_states_follow_set</i>	13	7
<i>prt_common_follow_set_la</i>	14	8
Cprt_xrefs_docs user-prefix-declaration directive	15	8
<i>Rprt_xrefs_docs</i> rule	16	9
<i>Rrules</i> rule	17	11
<i>Rrule</i> rule	18	12
<i>Rrule_def</i> rule	19	12
<i>Rsubrules</i> rule	20	12
<i>Rsubrule</i> rule	21	12
<i>Rsubrule_def</i> rule	22	12
<i>Relements</i> rule	23	12
<i>Relement</i> rule	24	13
<i>Relement</i> 's subrule 1	25	13
<i>Relement</i> 's subrule 2	26	13
<i>Relement</i> 's subrule 3	27	14
<i>Relement</i> 's subrule 4	28	14
<i>Relement</i> 's subrule 5	29	15
<i>Relement</i> 's subrule 6	30	15
First Set Language for O_2^{linker}	31	16
Lr1 State Network	32	17
Index	33	20