Table of Contents

总览 1.1

板形主流程 1.2

分配计算 1.2.1

SSU日志验算 1.3

SSU日志FAQ 1.4

CFG模型参数的梳理 1.5

板形模型消化

Go！！！！！

cShapeSetupD::Main(..)

`cShapeSetupD` 对象有一个 `status` 状态量，显示当前板形设定的状态（红灯或绿灯），以及判断是否合法的指示器 `ok` 。初始默认情况下 `status` 设为红灯， `ok` 设为 `false` 。

```
this->status = cMdlparam::cs_red;
this->ok     = false;
```

`roll_change_count` 是判断是否换辊的计数器。

```
int    roll_change_count (0);
```

`redrft_perm` 用于判断是否可以重新分配各机架厚度或者重新分配压下。

初始情况下 `redrft_perm` 设为*false*。

后根据如下条件更新 `redrft_perm` 的值。*ssu_load_enab*一般为*false*。*ssu_granted*一般为*true*。*s_CalId*指的是*FCD*对象中的*Calculation ID*，最小为*1*，最大为*2*。

```
redrft_perm =   ( true  == pcSched->pcSetupD->pcSetup->ssu_load_enab ) &&
  ( true  == pcSched->pcFSSched->pcSSys->state.ssu_granted ) &&
  ( 1 == pcSched->pcFCD->state.s_CalId );
```

`redrft_perm` 的值与 `iter` 相同。

```
    if ( redrft_perm )
    {
        iter = 1;
    }
    else
    {
        iter = 0;
    }
```

在日志中的*1st.iter*与源码中此处 `iter` 并不一样。

短期自学习

短期自学习的设定受到换辊和钢种、规格跳档的影响。

跳档

针对钢种和规格的跳档，模型考虑了以下五种情况。

```
        this->family_chg         = false;
        this->narrow_to_wide_chg = false;
        this->wide_to_narrow_chg = false;
        this->prd_chg            = false;
        this->lot_chg            = false;
```

钢种族跳档、宽度由窄变宽、宽度由宽变窄，这些都好理解。

而*prd_chg*指的是：钢种族跳档、宽度由窄变宽、宽度由宽变窄这三种情况至少有一种出现。

```
        if ( (true == this->family_chg)         ||
             (true == this->narrow_to_wide_chg) ||
             (true == this->wide_to_narrow_chg) )
        {
            this->prd_chg = true;
        }
```

*lot_chg*指的是和前一块带钢相比，钢种族、厚度索引、宽度索引其中至少一者发生改变，则称为*lot_chg*。

```
if (  // family change
    ((pcSched->pcFSSched->pcSAMP->state.pr_family > 0) &&
    (abs(pcSched->pcPDI->state.family - pcSched->pcFSSched->pcSAMP->state.pr_fami
ly) > 0)) ||
    // gauge range table index change
    ((pcSched->pcFSSched->pcSAMP->state.pr_grt_idx > 0) &&
    (abs(pcSched->pcPDI->state.grt_idx - pcSched->pcFSSched->pcSAMP->state.pr_grt
_idx) > 0)) ||
    // width range table index change
    ((pcSched->pcFSSched->pcSAMP->state.pr_wrt_idx > 0) &&
    (abs(pcSched->pcPDI->state.wrt_idx - pcSched->pcFSSched->pcSAMP->state.pr_wrt
_idx) > 0)) )
{
    this->lot_chg = true;
}
```

短期自学习预设定

根据是否跳档以及换辊，短期自学习在开始进行设定设定计算前，更新凸度和平直度的自学习。所谓的预设定其实是在不同的条件下清零自学习。

- 当有机架换辊，则累积增加相应机架的换辊次数*num_rolls_chgd*。
- 若超过两个机架换辊，则清零凸度自学习与平直度自学习（*vrn*和*err*），并更新辊形自学习，同时更换轧辊的相应道次清零弯辊力补偿。
- 当出现*lot_chg*，清零弯辊力补偿以及*bnd_ofs_counter*。
- 当出现钢种族跳档，则清零凸度自学习（*vrn*和*err*）。
- 如果出现窄到宽的跳档，同时周期内轧制块数超过*30*块，并且存在中浪趋势，则清零平直度自学习（*vrn*和*err*），若辊形自学习也小于零，则清零辊形自学习*wr_crn_vrn*。
- 若出现宽到窄的跳档，不做任何设定修改。

更新完短期自学习表之后，并且当前计算阶段处于*course-2*之后，则*put*短期自学习表到模型数据库（*models database*）。

cShapeSetupD::Init(..)

*cShapeSetupD::Init(..)*初始化了动态的*SHAPESETUP*对象以及其它相关的动态对象，比如：*LPCE*、*LRG*、*UFD*和*TARGT*。除此之外，这个函数还计算了执行机构的软极限，同时复制外部的数据给合适的动态对象。

*cShapeSetupD::Init(..)*初始化之后，最初的哪两个状态布尔值更新为*true*。

```
this->ok     = true;
this->status = cMdlparam::cs_green;
```

*cShapeSetupD::Init(..)*的实现在*shapesetup_req.cxx*文件中。

长短期自学习初始化

首先初始化凸度和平直度的目标*tgt_profile*和*tgt_flatness*。这两个目标一开始是*PDI*目标加上操作工的补偿。

模型用*prf_vrn_sel_flag*和*flt_vrn_sel_flag*这两个参数来标识长短期自学习的选择，默认以长期自学习为主。初始的凸度或平直度自学习为长期自学习，当这一块带钢和上一块带钢相比，出现钢种或规格跳档，则将长期自学习加上上一块增益后的短期自学习，作为新的凸度自学习*prf_vrn_rm_tmp*和*prf_vrn_rs_tmp*；以及平直度自学习*flt_vrn_tmp*。

cTargtD::Init(..)

*cTargtD::Init(..)*中主要确定初始的凸度以及目标有效凸度的极限。*prf_vrn*是*prf_vrn_rm_tmp*和*prf_vrn_rs_tmp*的差，*flt_vrn*就是*flt_vrn_tmp*。

目标*flt*为*pdi*平直度目标加上平直度的操作工补偿。

初始目标凸度与平直度稍有差别。

```
prf_int = (pdi_prf + prf_op_off) * matl_exp_cof + prf_vrn;
```

*prf_int*为*pdi*凸度加上操作工补偿后的热态凸度，再加上凸度自学习量。也就是说，凸度自学习量是补偿热态下的凸度。

之后用凸度的容许偏差计算单位凸度的上下极限。

初始化的大循环

cShapeSetupD::References(..)

*cShapeSetupD::References(..)*计算了凸度与平直度控制目标下的相关设定值，必要情况下重新分配轧制力或压下。

cAlcD::Calculate(..)

凸度分配计算。

Delvry_Pass(..)之前

cAlcD::Calculate(..)开始时，首先赋值中间坯的“分配厚度”。中间坯的“分配厚度”实际为F1的入口厚度，即中间坯的实际厚度。接着计算F1到F7的单位轧制力、分配厚度，以及引用轧辊咬入相关的对象。

如果可以重新分配压下，那么还会计算轧制力的最大改变量。

之后计算总的单位凸度改变量pu_prf_change_sum。

```
pu_prf_change_sum +=
  pcFSPassDtmp->pcEvlLPceD[ iter ]->strn_rlf_cof
  / (pcFSPassDtmp->pcFSStdD[ iter ]->pcLRGD->pce_infl_cof
  * pcFSPassDtmp->pcEvlLPceD[ iter ]->elas_modu);
```

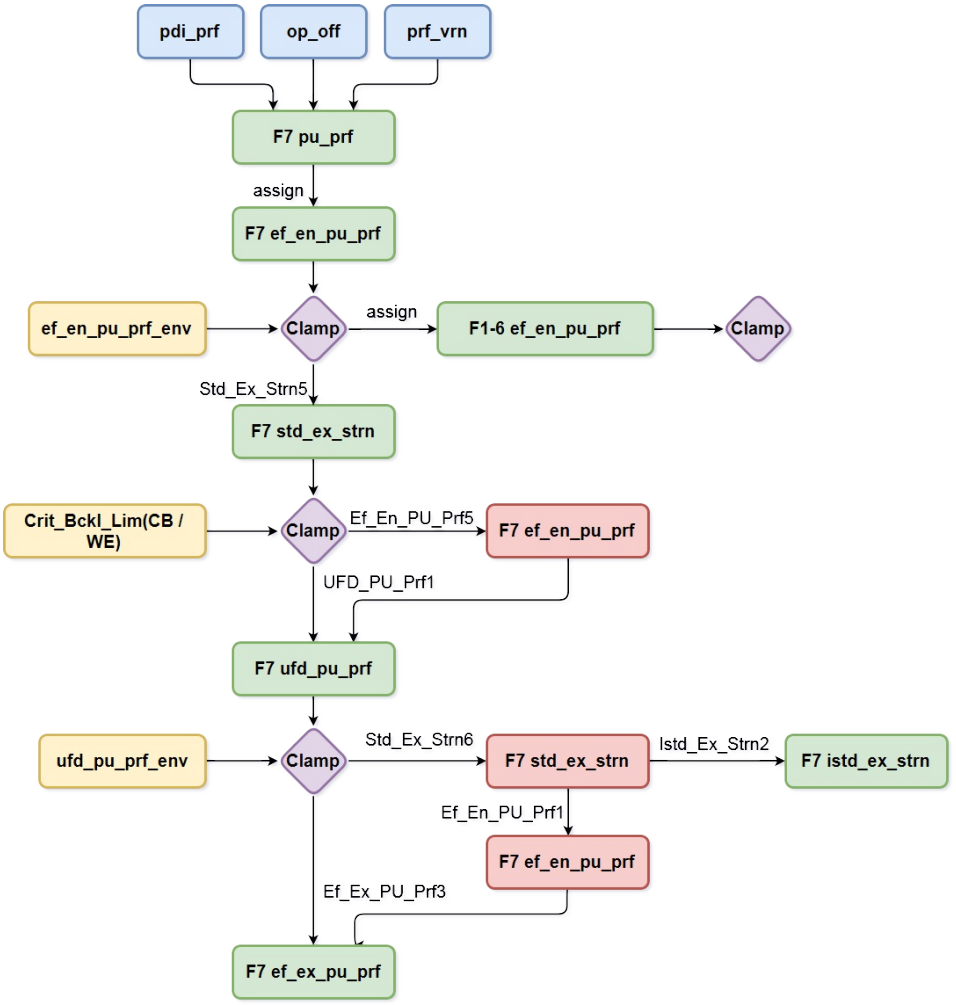
或者表示为：

$$pu\_prf\_change\_sum = \sum_{i=1}^{7}(\frac{strn\_rlf\_cof}{pce\_infl\_cof} \cdot Q_{elas\_modu})$$

Delvry_Pass(..)

Delvry_Pass(..)计算F7或最后一非空过道次的入口和出口有效单位凸度，以及出口istd应变差。

计算流程如下图所示。



基础量是加了操作工补偿和凸度自学习的单位凸度*pu_prf*。由*pu_prf_env*限幅。

首先将*pu_prf*直接赋值给最后一道次（*F7*）入口有效单位凸度*ef_en_pu_prf*。

之后将*F7*的*ef_en_pu_prf*直接赋值给*F1*到*F6*的入口有效单位凸度*ef_en_pu_prf*，并由各个道次的*ef_pu_prf_env*限幅。这是先假设所有机架单位凸度相同，理想状态下的情况。

回到*F7*，用*Std_Ex_strn5*计算末道次的机架出口应变差*std_ex_strn*，并用中浪和边浪的判别极限*Crit_Bckl_Lim*限幅。在判别极限中，中浪对应负值，边浪对应正值。限幅操作的意义是看*std_ex_strn*是否超出判别极限，若超出则一定会出现浪

形，则当前*std_ex_strn*的值肯定不合适，需要重新计算，但是*std_ex_strn*的值依赖*ef_en_pu_prf*。因此，通过函数*Ef_En_PU_Prf5*，利用限幅后的*std_ex_strn*和目标*pu_prf*重计算*F7*的*ef_en_pu_prf*。

接着利用新的*std_ex_strn*计算*F7*的*ufd_pu_prf_buf*。同样在限幅时，若发现当前的*ufd_pu_prf_buf*不合适，需要重新计算。但是*ufd_pu_prf*的计算依赖于之前的*std_ex_strn*、*ef_en_pu_prf*计算结果，因此必须对这两个值重新计算。

最后，我们有重新计算的*std_ex_strn*、*ef_en_pu_prf*值。即可通过新的*std_ex_strn*值，计算出*F7*的出口*istd*应变差。利用新的*ef_en_pu_prf*和*ufd_pu_prf_buf*计算出*F7*的出口有效单位凸度*ef_ex_pu_prf*。

注意在*Delvry_Pass(..)*中的均载辊缝凸度只是作为中间计算结果存在，与后面分配阶段的*ufd_pu_prf*有所区别。

凸度分配计算的大循环

前期准备工作做完后进入凸度分配计算的大循环。

局部指针的引用

在每个循环体开始执行时，先用局部指针指向本次循环要用到的所有相关动态对象。

```
// create pointers to class objects that are part of this pass
pcStdD    = pcFSPassD->pcFSStdD[ iter ];
pcCRLCD   = pcStdD->pcCRLCD;
pcAlcD    = pcFSPassD->pcAlcD;
pcUFDD    = pcStdD->pcUFDD;
pcLRGD    = pcStdD->pcLRGD;
pcLPceD   = pcFSPassD->pcLPceD;
pcPEnvD   = pcFSPassD->pcPEnvD;
pcEnPceD  = pcStdD->pcEnPceD;
pcExPceD  = pcStdD->pcExPceD;
// create a pointer to the previous active pass
pcPrvAct = pcFSPassD->pcPrvAct;
```

目的是为提高性能。

更新综合辊缝凸度

凸度方面，模型首先更新综合辊缝凸度，保证带钢–工作辊凸度*pce_wr_crn*和工作辊–支承辊凸度*wr_br_crn*是当前状态下的最新值。

```
//-----------------------------------------------------------------
// Calculate the following composite roll stack crown quantities:
//     Piece to work roll stack crown
//     Work roll to backup roll stack crown
//-----------------------------------------------------------------
line_num = __LINE__;
pcCRLCD->Crns ( pcStdD->wr_shft,
                pcStdD->angl_pc,
                pce_wr_crn,
                wr_br_crn );
```

*pcCRLCD–>Crns(..)*的计算详见*CRLC*模块说明。

空过的分配处理

分配从*F7*或末道次机架，从后往前倒者来。

若非末道次机架中，若本道次空过，则则传递本道次的出口厚度给上游机架，也就是空过的机架前后带钢厚度不变。并且设定本道次均载辊缝凸度*ufd_pu_prf*为*0*。

```
            if( pcStdD->dummied )
            {
                //-----------------------------------------------------------------
                // 保存带钢的出口厚度给入口厚度.
                //-----------------------------------------------------------------
                ( ( cFSPassD* )pcFSPassD->previous_obj )->pcAlcD->thick = pcAlcD->
thick;
                pcAlcD->ufd_pu_prf = 0.0;
            }
```

非空过部分的计算持续到*start_over*之前。

咬入计算与单位轧制力约束

之后进行带钢的咬入计算，咬入计算的输入量有入口宽度、出入口厚度、出入口张力、轧制速度，计算输出量有单位轧制力、前滑值和接触弧长度。

```
pcAlcD->pcRollbite->Calculate(     //@S014
                    &rbStatus,                          // OUT status from calcul
ations
                    &force_pu_wid_buf,                  // OUT rolling force/width

                    &fwd_slip,                          // OUT exit slip ratio [-]

                    &arcon,                             // OUT length of arc [min
or_length]
                    ( ( cFSPassD* )pcFSPassD->previous_obj )->pcAlcD->thick,  //
IN  entry_thk
                    pcAlcD->thick,                      // IN  exit_thk
                    pcEnPceD->width,                    // IN  exit/entry width
                    pcStdD->speed,                      // IN  roll peripheral sp
eed
                    pcEnPceD->tension,                  // IN Entry tension
                    pcExPceD->tension )                 // IN Exit tension
```

为什么把咬入计算放在这里，是因为后面有重分配压下的打算，即*redrft_perm*为*true*时，需约束单位轧制力。

```
            if( redrft_perm )
            {
                pcAlcD->force_pu_wid = (float) force_pu_wid_buf;

                //-------------------------------------------------------------
                // Restrict the rolling force per unit piece width to within
                // the rolling force per unit piece width envelope.
                //-------------------------------------------------------------
                line_num = __LINE__;
                cAlcD::Eval_Frc_PU_Wid( force_pu_wid_clp,
                                        pcAlcD->force_pu_wid,
                                        pcStdD->force_strip / pcStdD->pcEnPceD->w
idth,
                                        pcPEnvD->force_pu_wid_env,
                                        pcAlcD->pcRollbite->Precision() );
            }
            pcAlcD->flt_ok = true;
```

约束完单位轧制力后，设定一个标识浪形是否合格的指示器：*pcAlcD->flt_ok*，其默认值为*true*。

均载辊缝单位凸度的计算

最重要的计算到来了，均载辊缝单位凸度的计算。

注意这里有两个均载辊缝单位凸度，一个是*pcLPceD->ufd_pu_prf*，另一个是*pcAlcD->ufd_pu_prf*，这两个*ufd_pu_prf*是相对的，因为要比较它们之间的偏差。

如果下游机架的带钢影响系数为*0*，则只计算实际的*pcAlcD->ufd_pu_prf*，不更新*pcLPceD->ufd_pu_prf*。如果下游机架的带钢影响系数不为*0*，则计算目标均载辊缝单位凸度*pcAlcD->ufd_pu_prf*，接着计算弯窜辊，最后用*18*项线性方程更新*pcLPceD->ufd_pu_prf*。

计算弯窜辊过程中可以选择执行机构的计算先后顺序，目前是先计算窜辊，再计算弯辊。执行机构的计算顺序保存在*actr_prior*中。*cAlc::actrtyp_shift*的条件则进行窜辊位置的计算，*cAlc::actrtyp_bend*的条件则进行弯辊力的计算。注意在优先级别*actr_prior*中*cAlc::actrtyp_none*，指的是无执行机构执行计算，表示预设位。

在窜辊计算中，首先根据目标均载辊缝单位凸度*pcAlcD->ufd_pu_prf*利用*18*项线性方程反算综合辊缝凸度：带钢-工作辊凸度*pce_wr_crn*和工作辊-支承辊凸度*wr_br_crn*。之后*pce_wr_crn*代入*pcCRLCD->Shft_Pos(..)*计算窜辊位置，最后利用*pcCRLCD-> Crns(..)*和新计算的窜弯辊值更新综合辊缝凸度。

在窜辊计算中，用*pcUFDD->Bnd_Frc(..)*反算弯辊力，注意输出量*force_bnd*为*force_bnd_des*限幅后的结果。

设定一个表示目标均载辊缝单位凸度和实际均载辊缝单位凸度偏差的指示器。若偏差大于*ufd_pu_prf_tol*（目前为*0.0001*）则设定为*true*，表示均载辊缝单位凸度偏差超出了容许的范围，引出了后面有关*alc_lim*缩小偏差的一系列计算。

```
alc_lim = fabs( pcAlcD->ufd_pu_prf - pcLPceD->ufd_pu_prf ) > pcAlcD->pcAlc->ufd_p
u_prf_tol;
```

大循环中*redrft_perm*相关计算

重算单位轧制力以及更新相关动态参数。由于*redrft_perm*为*false*，这里的计算内容忽略。

alc_lim相关计算

首先计算个局部变量的ufd_pu_prf，用pcUFDD->Prf(..)计算，注意若轧制力不允许重新分配，那么这个局部的ufd_pu_prf和pcLPceD->ufd_pu_prf（目标）是一样的。也就是说，若轧制力重新分配，需要使用新的弯辊力和综合凸度去更新ufd_pu_prf。

接着利用pcLRGD->Ef_En_PU_Prf3(..)计算新的入口有效单位凸度ef_en_pu_prf_buf。此时旧的入口有效单位凸度为ef_en_pu_prf（old）。

上一道次的有效单位凸度包络线当然可以约束ef_en_pu_prf_buf。但是这种约束并不准确，因为厚度可能会变，因此约束标准应当有所放宽。所以在程序中用出口有效单位凸度ef_ex_pu_prf来约束。约束后新的入口有效单位凸度设为ef_en_pu_prf。

用新的ef_en_pu_prf求出std_ex_strn和ef_ex_pu_prf。至此，在这个阶段我们获得了可能合适的出入口有效单位凸度。但是别急，还需要判断出口的浪形，才能决定我们目前的分配凸度是否合适。

分配模型中，如果进入了alc_lim的计算，在重计算出入口有效单位凸度后必须进行浪形判别。F1和F6本道次的浪形判别，由本道次的应变差和下一道次应变差是否超死区极限决定。F7道次的浪形判别，仅由本道次F7的应变差是否超限决定。

当浪形判别不通过，或者说flt_ok为假时，可以稍微放宽一点标准。如果非末道次机架的下道次应变差不超死区极限，那么也算本道次浪形判别通过。

```
if ( pcFSPassD != pcLstActFSPassD )
{
  pcAlcD->flt_ok =
    ( std_ex_strn_dn <= bckl_lim_dn[ we ] ) &&
    ( std_ex_strn_dn >= bckl_lim_dn[ cb ] );
}
```

到现在这一步，如果浪形判别还不能通过，那么需要重新设定目标单位凸度。先重新设定目标有效单位凸度，用pcTargtD->Pass_Mill_Targ(..)计算获得，变量为ef_pu_prf_alt。再利用F7的istd_ex_strn反推std_ex_strn，结合ef_pu_prf_alt计

算出目标单位凸度*pu_prf*。注意在*alc*模块中，大循环的这个位置是*alc*局部*pu_prf*变量第一次介入的地方，局部*pu_prf*变量预设为*0*。之后对*pu_prf*进行限幅，这样新的目标单位凸度就诞生了。

接着是一个难点问题。

```
pcCritFSPassD = pcFSPassD;
```

将更新了目标单位凸度的道次地址赋值给*pcCritFSPassD*指针。这个*pcCritFSPassD*最开始是指向*F7*道次的。*pcCritFSPassD*指针设定的意义在于：在迭代计算的过程中，浪形判别不合格的相应道次必须比之前更新过目标单位凸度的道次低。

在*alc_lim*计算过程的最后，若目标单位凸度发生改变，则设定*start_over*指示器为*true*，以进行后续*start_over*的流程。

非空过道次的更新

在*alc_lim*计算之后，更新给定条件下的*ufd_pu_prf*、*ef_pu_prf*、*strn*、*prf*，注意这些值都属于*Ipce*对象。

*start_over*流程

如果目标单位凸度发生改变，则更新目标单位凸度的迭代次数（累积加一）。之后从*F7*重新开始大循环的计算，从*Delivry_Pass(..)*起步重算出入口有效单位凸度，并设定*F7*的有效单位凸度为出口有效凸度。

如果目标单位凸度没有发生改变，说明本道次的浪形是符合判别条件的，不需要更改目标；或者目标均载辊缝凸度达到了实际的均载辊缝凸度。进一步说，有两种情况会进入*start_over*为假的流程，一种是未进入*alc_lim*计算的状态，另一种是进入了*alc_lim*的计算，但是浪形判别合格的状态。则当前道次对象*pcFSPassD*可以前移一个道次。出入口有效单位凸度现在敲定是合适的。

之后是：对不均匀变形道次*ef_en_pu_prf*修正的过程。在此阶段，如下一段代码需要注意，在理解上可能会出错。

```
//-----------------------------------------------------
// Increment pointer to previous dynamic PASS object.
```

```
//-----------------------------------------------------
pcFSPassD = ( cFSPassD* )pcFSPassD->previous_obj;

//-------------------------------------------------------------
// Save the effective entry per unit profile of the previous
// pass into the effective exit per unit profile for this pass.
//-------------------------------------------------------------
ef_ex_pu_prf = ef_en_pu_prf;

//-------------------------------------------------------------
// Determine the upstream effective per unit profile to aim
// towards using the extreme downstream pass where the piece
// influence coefficient is zero.
//-------------------------------------------------------------
ef_en_pu_prf = cMathUty::Clamp ( ef_ex_pu_prf,
    pcPceIZFSPassD->pcPEnvD->ef_pu_prf_env[ minl ],
    pcPceIZFSPassD->pcPEnvD->ef_pu_prf_env[ maxl ] );
```

这段代码执行之后，*ef_ex_pu_prf*这个变量的意义已经发生改变，不再代表本道次的出口有效单位凸度。因为前面我们的*pcFSPassD*道次对象已经前移一个道次，因此程序的设计者为了简练，直接使用局部变量*ef_en_pu_prf*代表其它含义。

如代码所示，*ef_ex_pu_prf*保存的是原*ef_en_pu_prf*的值，而新的*ef_en_pu_prf*是受到*pcPceIZFSPassD*道次*ef_pu_prf_env*包络线限幅之后的值。

没经过浪形判别或经过浪形判别但没有改变目标单位凸度的过程参数*ef_en_pu_prf*，必须从出现不均匀变形的机架开始，用每个机架的单位凸度最大改变量约束和修正本道次的*ef_en_pu_prf*。*ef_en_pu_prf*修正过程是个循环，从*pcPceIZFSPassD*道次的下一道次开始，且当前道次的上一道次在不均匀延伸的机架中，直至末道次。

*ef_pu_prf_chg[cb/we]*并不能直接作为真正的有效单位凸度最大改变量，或真正的有效凸度改变约束条件。在不均匀变形的机架中，它需要本道次的*ef_pu_prf_env*和上一道次的*ef_pu_prf_env*介入，来获得一个更窄的变化区间*ef_pu_prf_dlt[minl/maxl]*，用所有存在不均匀变形机架的这个区间来修正本道次的*ef_en_pu_prf*。

```
//-------------------------------------------------------
// Calculate the delta effective per unit profile change
// from stand entry to interstand exit.
//-------------------------------------------------------

ef_pu_prf_dlt[ minl ] =
```

```
    cMathUty::Max( pcBufFSPassD->pcFSStdD[ iter]->pcLRGD->ef_pu_prf_chg[ cb ],
        cMathUty::Max( ef_ex_pu_prf,
                       pcBufFSPassD->pcPEnvD->ef_pu_prf_env[ minl ] ) -
        cMathUty::Min( ef_en_pu_prf,
                   ((cFSPassD*)pcBufFSPassD->previous_obj)->pcPEnvD->ef_pu_prf_e
nv[maxl]));

ef_pu_prf_dlt[ maxl ] =
    cMathUty::Min( pcBufFSPassD->pcFSStdD[ iter]->pcLRGD->ef_pu_prf_chg[ we ],
        cMathUty::Min( ef_ex_pu_prf,
                       pcBufFSPassD->pcPEnvD->ef_pu_prf_env[ maxl ] ) -
        cMathUty::Max( ef_en_pu_prf,
                       ((cFSPassD*)pcBufFSPassD->previous_obj)->pcPEnvD->ef_pu_p
rf_env[minl]));
```

*ef_pu_prf_dlt*的计算，讲白了就是用本道次机架的有效凸度减前一道次机架的有效凸度，只不过将本道次机架和前道次机架的包络线和*ef_ex_pu_prf*、*ef_en_pu_prf*联系起来，用于收窄死区。*ef_pu_prf_chg*依据于理论计算，第二项的差值依据于本道次和上道次的包络线，若中浪则取最大的，若边浪则取最小的。

先求有效凸度改变的总量。从出现不均匀变形的机架（*pcPcelZFSPassD*道次的下一道次）开始，如果*pcPcelZFSPassD*道次包络线限幅后的*ef_en_pu_prf*小于原值，说明*pcPcelZFSPassD*道次的包络线区间整体小于原*ef_en_pu_prf*，那么*ef_pu_prf_sum*累加*ef_pu_prf_dlt[maxl]*。反之，若*pcPcelZFSPassD*道次包络线限幅后的*ef_en_pu_prf*大于原值，说明*pcPcelZFSPassD*道次的包络线区间整体小于原*ef_en_pu_prf*，那么*ef_pu_prf_sum*累加*ef_pu_prf_dlt[minl]*。

注意*pcFSPassD*是本道次的前一道次，而在修正*ef_en_pu_prf*的循环当中，*pcBufFSPassD*指向本道次的前一道次时，循环结束。

这时*ef_en_pu_prf*变量的含义又发生了改变，变回了字面意思，即本道次的入口有效单位凸度。修正是从原始初设定的入口有效单位凸度加上（或减去）与累加有效凸度改变总量成比例的一部分*ef_pu_prf_dlt*，作为新的入口有效单位凸度存在。

```
// 以边浪情形为例
ef_en_pu_prf =
    ef_ex_pu_prf - ef_pu_prf_dlt[ maxl ] *
    ( ef_ex_pu_prf - ef_en_pu_prf ) /
    ef_pu_prf_sum;
```

修正结束后，考虑*ef_en_pu_prf*和*ef_ex_pu_prf*偏差太大的情况，则计算*ef_en_pu_prf_dft*作为最终的*ef_en_pu_prf*。

```
float ef_en_pu_prf_dft = ef_ex_pu_prf + (pcTargtD->en_pu_prf - pcTargtD->pu_prf)
        * pcFSPassD->pcAlcD->pu_prf_change;
```

最后用*pcTargtD->Eval_Ef_En_PU_Prf(..)*评估一下*ef_en_pu_prf*和*ef_ex_pu_prf*。

后续其它计算

若轧制力不重新分配，则预设所有机架*force_ssu*为*0*。

SSU日志验算

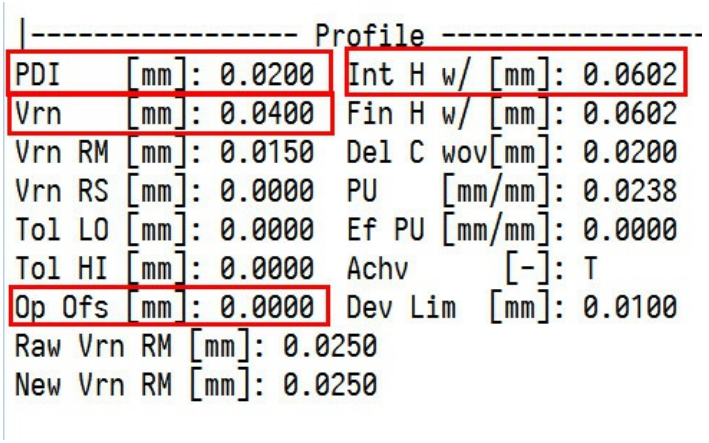
日志验算和说明。

Int H w和Fin H w

--- Profile ---这一栏当中的 “H w” 指的是加了凸度自学习量的目标凸度。Int H w为初始的prf，Int H w为最终的prf。Int H w的计算过程如下所示。

```
prf_int = (pdi_prf + prf_op_off) * matl_exp_cof + prf_vrn;
```

热膨胀系数可以忽略不计，那么Int H w就是PDI的凸度、操作工补偿的凸度、凸度自学习的和。



在板形模型目标初始化阶段，prf_vrn会被赋值为prf_vrn_rm和prf_vrn_rs的差（prf_vrn_rs常年为零）。从实际数据来看，prf_vrn_rm和prf_vrn_rs的差，与Vrn还是存在差距的，说明这里的Vrn在实际计算中还会出现变化。（可能存在板形问题？）

中间坯凸度插值计算

中间坯的凸度，在模型计算中不是

中间坯有效单位凸度

中间坯不存在应变差的概念，中间坯的单位凸度就是中间坯的有效单位凸度。

*--- Transfer Bar ---*中有每卷带钢的中间坯凸度*Prof*，由模型插值计算获得；以及中间坯的厚度，*R2*出口测量获得。可以通过凸度求厚度的商，作为中间坯的有效单位凸度，其值与*--- Allocation Results (1st Iter.) ---*这一栏中的零道次*pass0*的*EF PU Prf*对应。中间坯凸度值与*--- Allocation Results (1st Iter.) ---*这一栏中的零道次*pass0*的*Prf*值对应。

|------- Allocation Results (1st Iter.) ------|

	Des Bnd Frc	Ef PU Prf		UFD PU Prf	Std Ex Strn
#	[kN]	[mm/mm]	[mm]	[mm/mm]	[mm/mm]
0		0.0063	0.2865		
1	815	0.0214	0.4122	0.0239	0.00017
2	890	0.0234	0.2100	0.0268	0.00014
3	890	0.0236	0.1417	0.0272	0.00016
4	890	0.0238	0.1026	0.0287	0.00018
5	840	0.0238	0.0814	0.0305	0.00021
6	315	0.0239	0.0686	0.0297	0.00018
7	320	0.0239	0.0602	0.0212	-0.00009

单位宽度轧制力的验算

单位宽度轧制力，直接拿轧制力除以带钢精轧宽度，由于受到限幅，一般单位轧制力的值是真正被限幅后的值。注意这里的精轧宽度指的是中间坯在*F1*入口的宽度，不是订单宽度也不是目标宽度。

所以，单位轧制力包络线的最大值、单位轧制力包络线的最小值，以及*--Allocation Requirements (1st Iter.) --*这一栏中的单位轧制力分配结果一般三者在各自机架内都相等。

带钢弹性模量的计算误差

手动验算的时候发现*elas_mod*的计算结果总是与*SSU*日志中的记录结果存在一定的偏差，偏差精度在*100*以内。

产生这样偏差的原因是因为插值。弹性模量的插值通过每个机架的温度完成。插值表如下表所示。

avg_pce_tmp_interp_vec	elas_modu_interp_vec
600	138269
650	128069
700	117905
750	107751
800	97589
850	87415
900	77232
950	67054
1000	56909
1050	46829
1100	36863
1500	27067

带钢弹性模量的插值向量从27067到138269MPa，与温度的插值相比已经不在一个数量级。温度一单位改变会造成弹性模量更大的变动，因此出现100MPa以内的波动很正常。

SSU日志 FAQ

Profile一栏中Vrn RM和Vrn RS是什么？

Profile一栏中Vrn RM和Vrn RS一般情况下指的是长期自学习值。

pcTargtD->pcTargt->prf_vrn_sel_flag默认值为true，在cfg_targt.txt文件中设定，若此值为false，则Profile一栏中Vrn RM和Vrn RS为短期自学习的psSAMP->prf_vrn_rm和psSAMP->prf_vrn_rs。

为什么弯辊力包络线最大值和最小值相反？

标签max和min指的是ufd有效单位凸度的最大值和最小值，最大的弯辊力会计算获得最小的有效单位凸度，最小的弯辊力会计算获得最大的有效单位凸度。为保持一致性，弯辊力包络线max与min对调。

wr_crn_vrn_z是什么？

算是凸度自学习的一个初始值。当换辊算不准时restore进行补偿。配置文件里面有，据说很好用。

CFG参数整理

TARGT

```
class = cTargt;

   cTargt = targt;

       flt_err_lim     = 2,                    ! [kN] flatness errorlimits

                           -250.,               !    minimum

                            250.;               !    maximum
    // 平直度自学习当中的平直度偏差极限范围

       flt_vrn_bled    = 0.9;                  ! [-] target flatness vernierbleed-
off
    // 平直度自学习衰减系数

       flt_vrn_lim     = 2,                    ! [kN] target flatnessvernier limi
ts

                           -800.0,              !    minimum

                            800.0;              !    maximum

    // 平直度自学习极限范围
      flt_vrn_i_gn    = 0.6;                  ! [-] target flatness controlloop i
ntegral gain

       flt_vrn_p_gn    = 0.3;                  ! [-] target flatness controlloop
proportional gain
    // 平直度自学习PI控制系数

       prf_dev_lim     = 0.010;                ! [mm] target profile deviationlim
it
    // 凸度波动极限

       prf_err_lim     = 2,                    ! [mm] profile error limits

                           -0.100,              !    minimum

                            0.100;              !    maximum
    // 凸度自学习偏差极限

       prf_lim         = 2,                    ! [mm] absolute limits
```

```
                          0.000,                ! minimum
                          0.250;                ! maximum
    // 凸度最大范围
       prf_tol         = 2,                    ! [mm] target profiletolerances
                         -0.050,                ! minimum
                          0.050;                ! maximum
    // 凸度精度要求
       prf_vrn_bled    = 0.9;                  ! [-] target profile vernier (re-p
redicted -setup) bleed-off
    // 凸度自学习衰减系数
       prf_vrn_lim     = 2,                    ! [mm] target profilevernier limit
s
                         -0.070,                ! minimum
                          0.070;                ! maximum
    // 凸度自学习极限
       prf_vrn_rm_i_gn = 0.4;                  ! [-] target profile vernier (re-pr
edicted - measured) control loop
                                                ! integral gain
       prf_vrn_rm_p_gn = 0.2;                  ! [-] target profile vernier (re-pr
edicted - measured) control loop
    // 凸度自学习rm的PI控制系数
                                                ! proportional gain
       prf_vrn_rs_i_gn = 0.2;                  ! [-] target profile vernier (re-pr
edicted - setup) control loop
    // 凸度自学习rs的PI控制系数
                                                ! integral gain
       flt_err_thrshld  = 100;                 ! [kN] Flatness errorthreshold min
imum for flatness feedback
    // 平直度自学习，学习的临界点
       opr_mx_wrng_corr = 100;                 ! [kN] Maximum operator correction
in WRONG direction and still doflatness feedback
    // 操作工如果调整错误，但仍然进行自学习的弯辊力临界点
       apc_start_std    = 1;                   ! [-] APC Correction startstand
    // APC修正开始的机架
       en_ex_strn_calc = true;                 ! [-] Enable exit strain calculatio
n
    // 是否允许计算出口应变差
       ex_strn_thk = 10.00;                    ! [mm] Exit strain match forthinkn
ess less than or equal to.
    // 出口应变差对应的最大厚度
       exclude_stainless = false;              ! [-] Exclude stainless steel
    // 是否排除不锈钢钢种
```

```
!@(CC087) start

       prf_vrn_sel_flag  = true;              ! [-] Vernier selection flag(false=
samp, true=slfg)
    // 凸度自学习，长短期以哪个为主的标识，默认长期自学习
       flt_vrn_sel_flag  = true;              ! [-] Vernier selection flag(false=
samp, true=slfg)
    // 平直度自学习，长短期以哪个为主的标识，默认长期自学习
!@(CC087) end

!@2ND(LC060) start

       wr_crn_off_sel_flag = true;           ! [-] Work roll offset selection fla
t (false= slfg, true = slfg+sprp)
    // 工作辊凸度补偿，标识是否使用sprp数据，默认使用长期自学习和sprp
!@2ND(LC061) end

   end;

end;
```