| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [FixedModeFlow/src](http://docs.google.com/index.html) - FixedModeShotDistributor.cpp (source / [functions](http://docs.google.com/FixedModeShotDistributor.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 69 | 96 | 71.9 % | | Date: | 2012-03-23 |  | Functions: | 7 | 10 | 70.0 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 7 | 22 | 31.8 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : #include "FixedModeShotDistributor.h"  2 : : #include "SFTable.h"  3 : : #include "SimplePattern.h"  4 : : #include "SlidingWindowPositions.h"  5 : : #include "SFLayer.h"  6 : : #include "TFOrderingPriority.h"  7 : : #include "TFRecycler.h"  8 : : #include "SDDB.h"  9 : : #include "print.h"  10 : :   11 : : namespace nft {  12 : :   13 : 2430 : boost::mutex FixedModeShotDistributor::mMutex;  14 : :   15 : : FixedModeShotDistributor::FixedModeShotDistributor(  16 : : const ShotTimeCalculator & inShotTimeCalc,  17 : : SFTable::Pointer inSFTable,   18 : : ProcessingMode inProcessingMode,  19 : : ConverterParams \*inConvParams )  20 : 1096 : : mSFTable( inSFTable ),  21 : : mShotTimeCalc( inShotTimeCalc ),  22 : : cMode( inProcessingMode ),  23 : : mParams( inConvParams ),  24 : : mAccessor( inSFTable, inConvParams ),  25 : : mTFLessModeDistributor( mSFTable ),  26 : 1096 : mPatBBox() {  27 : :   28 : 1096 : ConverterParams \*pParams = ConverterParams::instance();  29 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  30 : 1096 : = pParams->getStartupMessage();  31 : : LongPosition bbLLCorner( pStartMsg->llPtnProcessOriginX,  32 : 1096 : pStartMsg->llPtnProcessOriginY );  33 : : ULongPosition bbSize( pStartMsg->ullPtnProcessWidth,  34 : 1096 : pStartMsg->ullPtnProcessHeight );  35 : :   36 : 1096 : mPatBBox.setLowerLeftCorner( bbLLCorner );  37 : 1096 : mPatBBox.setSize( bbSize );  38 : : }  39 : :   40 : :   41 : 1096 : FixedModeShotDistributor::~FixedModeShotDistributor() {  42 : : }  43 : :   44 : : //Adds the shot after calculation into TF  45 : : void  46 : : FixedModeShotDistributor::operator()(  47 : : const SimplePattern & inDistPat,  48 : : const PatternPosition & inPosition,  49 : : uint32\_t inSFLNo,  50 : : ThreadID inThreadID,  51 : 17417 : uint32\_t inAI) throw(DPL::Error \*) {  52 : 17417 : SimplePattern inPat = inDistPat;  53 : 17417 : SDDB \*pSDDB = SDDB::instance();  54 : 17417 : e9k\_debug("Pattern at %lld %lld to be distributed\n",  55 : : inPosition.getX(), inPosition.getY());  56 : : #ifndef KEEP\_ZERO\_SIZE\_SHOT  57 : 17417 : if( 0 == inPat.getSizeX() || 0 == inPat.getSizeY() )  58 : : {  59 : :   60 : 1504 : try  61 : : {  62 : 1504 : pSDDB->shotDiscarded();  63 : : }  64 [ # # ]: 0 : catch(DPL::Error \*pChild)  65 : : {  66 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  67 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_,  68 : : \_\_LINE\_\_, ErrorHandler::eErrFunction,  69 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  70 : : "SDDB::shotDiscarded"));  71 : : }  72 : : //Zero pattern removed  73 : 1504 : return;  74 : : }  75 : : #endif //KEEP\_ZERO\_SIZE\_SHOT  76 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  77 : 15913 : = mParams->getStartupMessage();  78 : : LongPosition checkPos(  79 : : inPosition.getX() + pStartMsg->llDpbX,  80 : 15913 : inPosition.getY() + pStartMsg->llDpbY);  81 [ + + ]: 15913 : if( !mPatBBox.checkIsIncluded( checkPos ) )  82 : : {  83 : 93 : Logger::instance()->log(LogMessageTable::eSHOT\_INFO,  84 : : "Bounding box check failed for patterns");  85 : 93 : e9k\_debug("Pattern at %lld %lld discarded\n",  86 : : inPosition.getX(), inPosition.getY());  87 : 93 : try  88 : : {  89 : 93 : pSDDB->shotDiscarded();  90 : : }  91 [ # # ]: 0 : catch(DPL::Error \*pChild)  92 : : {  93 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  94 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_,  95 : : \_\_LINE\_\_, ErrorHandler::eErrFunction,  96 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  97 : : "SDDB::shotDiscarded"));  98 : : }  99 : : //TODO::LOG  100 : 93 : return;  101 : : }  102 : 15820 : try  103 : : {  104 : 15820 : pSDDB->SimplePatternAfterBBInclusion(inPat, inPosition);  105 : : }  106 [ # # ]: 0 : catch(DPL::Error \*pChild)  107 : : {  108 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  109 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_,  110 : : \_\_LINE\_\_, ErrorHandler::eErrFunction,  111 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  112 : : "SDDB::SimplePatternAfterBBInclusion"));  113 : : }  114 : :   115 [ + + ]: 15820 : if( eProcessTFLess == cMode )  116 : : {  117 : 67 : TF::Pointer tf = 0;  118 : 67 : try  119 : : {  120 : 67 : tf = TFRecycler::instance()->getTF();  121 : 67 : tf->setPatCode( inPat.getCode() );  122 : : }  123 [ # # ]: 0 : catch( DPL::Error \*pChild )  124 : : {  125 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  126 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  127 : : ErrorHandler::eErrFunction,  128 : : pErr->getErrorMsg(  129 : : ErrorHandler::eErrFunction,  130 : : "TFRecycler::getTF"));  131 : : }  132 : :   133 : 67 : try  134 : : {  135 : 67 : boost::mutex::scoped\_lock l(mMutex);  136 : 67 : TFInfo info;  137 : 67 : info.mTFPosition = inPosition;  138 : 67 : info.mTF = tf;  139 : 67 : info.mShotCode = inPat.getCode();  140 : 67 : info.mLayerNo = inSFLNo;  141 : 67 : mTFLessModeDistributor( info );  142 : 67 : if( tf->getDiscarded() )  143 : : {  144 : 0 : return;  145 : : }  146 : 67 : mShotTimeCalc.calculateShotTime(  147 : : inPat, inPosition, tf->getParentSF()->getDxpValue(), inSFLNo, inAI);  148 : : //inPat.setPositionX(0);  149 : : //inPat.setPositionY(0);  150 : 67 : tf->addShot( inPat, inThreadID );  151 : : }  152 [ # # ]: 0 : catch( DPL::Error \*pChild )  153 : : {  154 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  155 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  156 : : ErrorHandler::eErrFunction,  157 : : pErr->getErrorMsg(  158 : : ErrorHandler::eErrFunction, "TF::addShot"));  159 : : }  160 : : }  161 : : else  162 : : {  163 : 15753 : SFLayer::Pointer sfl = mSFTable->getSFLayers() + inSFLNo;  164 : 15753 : TFLayers::Pointer tfl = sfl->getTFL();  165 : 15753 : try  166 : : {  167 : 15753 : TF::Pointer tf = tfl->getTFForPattern( inPat, inPosition, inThreadID, inSFLNo);  168 [ + + ]: 15741 : if( !tf )  169 : : {  170 : 99 : try  171 : : {  172 : 99 : pSDDB->shotDiscarded();  173 : : }  174 [ # # ]: 0 : catch(DPL::Error \*pChild)  175 : : {  176 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  177 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_,  178 : : \_\_LINE\_\_, ErrorHandler::eErrFunction,  179 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  180 : : "SDDB::shotDiscarded"));  181 : : }  182 : : //Is only possible in delayed mode (not implemented)  183 : : //Or when pattern falls directly outside SFL  184 : 99 : return;  185 : : }  186 : 15642 : if( !tf->getDiscarded() )  187 : : {  188 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  189 : 15588 : = mParams->getStartupMessage();  190 : 15588 : mShotTimeCalc.calculateShotTime(  191 : : inPat, inPosition, tf->getParentSF()->getDxpValue(), inSFLNo, inAI);  192 : 15584 : inPat.setPositionX(  193 : : inPosition.getX()  194 : : - (tfl->getLayerLocationX()  195 : : + (tf->getColumn() \* pStartMsg->iTFSizeX)  196 : : + ( pStartMsg->iTFSizeX >> 1)));  197 : 15584 : inPat.setPositionY(  198 : : inPosition.getY()  199 : : - (tfl->getLayerLocationY()  200 : : + (tf->getRow() \* pStartMsg->iTFSizeY)  201 : : + ( pStartMsg->iTFSizeY >> 1)));  202 : 15584 : tf->addShot( inPat, inThreadID );  203 : 15584 : Logger::instance()->log(LogMessageTable::eSHOT\_POS,  204 : : inPosition.getX(), inPosition.getY(), tf->getRow(),  205 : : tf->getColumn());  206 : 15584 : e9k\_debug(  207 : : "Shot at position (%lld, %lld) added to TF(%d,%d)\n"  208 : : ,inPosition.getX(), inPosition.getY(), tf->getRow(),  209 : : tf->getColumn());  210 : : }  211 : : else  212 : : {  213 : 54 : Logger::instance()->log(LogMessageTable::eSHOT\_INFO,  214 : : "Pattern discarded as target TF falls out of DPB");  215 : : //Pattern discarded as target TF will finally fall out  216 : : //of DPB  217 : 54 : try  218 : : {  219 : 54 : pSDDB->shotDiscarded();  220 : : }  221 [ # # ]: 0 : catch(DPL::Error \*pChild)  222 : : {  223 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  224 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_,  225 : : \_\_LINE\_\_, ErrorHandler::eErrFunction,  226 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  227 : : "SDDB::shotDiscarded"));  228 : : }  229 : : }  230 : : }  231 [ + - ]: 16 : catch(DPL::Error \*pChild)  232 : : {  233 : 16 : e9k\_debug("ERROR OCCURRED");  234 : 16 : ErrorHandler \*pErr = ErrorHandler::instance();  235 : 16 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  236 : : ErrorHandler::eErrFunction,  237 : : pErr->getErrorMsg( ErrorHandler::eErrFunction,  238 : : "Shot time calculation or Shot addition to TF"));  239 : : //TODO::BREAKUP  240 : : }  241 : : }  242 : : }  243 : :   244 : : void  245 : : FixedModeShotDistributor::setShotTimeCalc(  246 : 0 : const ShotTimeCalculator &inValue) {  247 : 0 : mShotTimeCalc = inValue;  248 : : }  249 : :   250 : : void  251 : 0 : FixedModeShotDistributor::setSFTable(SFTable::Pointer inValue) {  252 : 0 : mSFTable = inValue;  253 : : }  254 : :   255 : : //Calculate shot time before adding to TF's  256 : :   257 : : void  258 : : FixedModeShotDistributor::calculateShotTime(  259 : 0 : const SimplePattern & pat) throw(DPL::Error \*){  260 : : }  261 : :   262 : :   263 : 1215 : } // namespace nft |

|  |
| --- |
| Generated by: [LCOV version 1.9](http://ltp.sourceforge.net/coverage/lcov.php) |