| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [SFLayers/src](http://docs.google.com/index.html) - SFTableAccessor.cpp (source / [functions](http://docs.google.com/SFTableAccessor.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 85 | 109 | 78.0 % | | Date: | 2012-03-23 |  | Functions: | 13 | 16 | 81.2 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 18 | 36 | 50.0 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : #include "ConverterParams.h"  2 : : #include "SFTableAccessor.h"  3 : : #include "TFOrderingPriority.h"  4 : : #include "TFLocationOrder.h"  5 : : #include "UtilTypes.h"  6 : :   7 : : #include "print.h"  8 : :   9 : : namespace nft {  10 : :   11 : 2430 : boost::mutex SFTableAccessor::mTFBufCreationLock;  12 : :   13 : : SFTableAccessor::SFTableAccessor(  14 : : SFTable::Pointer inSFTable, ConverterParams \*inParams )  15 : 11631 : :mSFTable( inSFTable ),  16 : 11631 : mParams( inParams ) {  17 : : }  18 : :   19 : 11631 : SFTableAccessor::~SFTableAccessor() {  20 : : }  21 : :   22 : :   23 : : SFTableAccessor::SFTableAccessor(  24 : : const SFTableAccessor & inSource)  25 : 0 : :mSFTable( inSource.mSFTable ),  26 : 0 : mParams( inSource.mParams ) {  27 : : }  28 : :   29 : : SFTableAccessor &  30 : 0 : SFTableAccessor::operator=(const SFTableAccessor & inSource) {  31 [ # # ]: 0 : if( this != &inSource )  32 : : {  33 : 0 : mSFTable = inSource.mSFTable;  34 : 0 : mParams = inSource.mParams;  35 : : }  36 : 0 : return \*this;  37 : : }  38 : :   39 : : SFLayer::Pointer  40 : 104084 : SFTableAccessor::getSFLayer(uint32\_t inSFLayerNo) const {  41 : :   42 : 52042 : if( mParams->getStartupMessage()->uiSFLCountInDpb <= inSFLayerNo )  43 : : {  44 : 0 : e9k\_debug("SFL not found for %u\n", inSFLayerNo);  45 : 0 : return NULL;  46 : : }  47 : 104084 : return mSFTable->getSFLayers() + inSFLayerNo;  48 : : }   49 : :   50 : : SFLayer::Pointer  51 : : SFTableAccessor::getSFLayer(  52 : 31314 : const LongPosition & inPosition, uint32\_t inSFLayerNo) const {  53 : 31314 : SFLayer::Pointer sfl = getSFLayer( inSFLayerNo );  54 [ - + ]: 31314 : if( NULL == sfl )  55 : : {  56 : 0 : return sfl;  57 : : }  58 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  59 : 31314 : = mParams->getStartupMessage();  60 : 31314 : int64\_t llx = sfl->getLLx();  61 : 31314 : int64\_t lly = sfl->getLLy();  62 : 31314 : if( inPosition.getX() < llx  63 : : || inPosition.getX() > llx +  64 : : sfl->getNoOfCols() \* pStartMsg->iSFSizeX ) {  65 : 72 : return NULL;  66 : : }  67 : 31242 : if( inPosition.getY() < lly  68 : : || inPosition.getY() > lly +  69 : : sfl->getNoOfRows() \* pStartMsg->iSFSizeY ) {  70 : 30 : return NULL;  71 : : }  72 : 31314 : return sfl;  73 : : }  74 : :   75 : : SFGroup::Pointer  76 : : SFTableAccessor::getSFGroup(  77 : 31314 : const LongPosition & inPosition, uint32\_t inSFLayerNo) const {  78 : 31314 : SFLayer::Pointer sfl = getSFLayer( inPosition, inSFLayerNo );  79 [ + + ]: 31314 : if( NULL == sfl )  80 : : {  81 : 102 : return NULL;  82 : : }  83 : :   84 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  85 : 31212 : = mParams->getStartupMessage();  86 : 31212 : int64\_t llx = sfl->getLLx();  87 : 31212 : int64\_t idx = (inPosition.getX() - llx)/pStartMsg->iSFSizeX;  88 : 31212 : if( idx > sfl->getNoOfCols() )  89 : : {  90 : 0 : return NULL;  91 : : }  92 : 31212 : idx = ( idx == sfl->getNoOfCols() ) ? idx - 1 : idx; //BOUNDARY  93 : 31212 : return sfl->getSFGroups() + idx;  94 : : }  95 : :   96 : : SubField::Pointer  97 : : SFTableAccessor::getSubField(  98 : 20779 : const LongPosition & inPosition, uint32\_t inSFLayerNo) const {  99 : 20779 : SFGroup::Pointer sfg = getSFGroup( inPosition, inSFLayerNo );  100 [ + + ]: 20779 : if( NULL == sfg )  101 : : {  102 : 51 : return NULL;  103 : : }  104 : 20728 : SFLayer::Pointer sfl = getSFLayer( inSFLayerNo );  105 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  106 : 20728 : = mParams->getStartupMessage();  107 : 20728 : int64\_t lly = sfl->getLLy();  108 : 20728 : int64\_t idy = (inPosition.getY() - lly)/pStartMsg->iSFSizeY;  109 : 20728 : idy = ( idy == sfg->getNoOfRows() ) ? idy - 1 : idy; //BOUNDARY  110 : 20728 : if( idy > sfl->getNoOfRows() )  111 : : {  112 : 0 : return NULL;  113 : : }  114 : 20728 : return sfg->getSubFields() + idy;  115 : : }  116 : :   117 : : Segment::Pointer  118 : : SFTableAccessor::getSegmentForLocationPriority(  119 : 8708 : const LongPosition & inPosition, uint32\_t inSFLayerNo) const {  120 : 8708 : Segment::Pointer pSegPointer=NULL;  121 : 8708 : try{  122 : 8708 : pSegPointer = calculateSegment( inPosition, inSFLayerNo, 0, true);  123 : : }  124 [ # # ]: 0 : catch( DPL::Error \*pChild )  125 : : {  126 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  127 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  128 : : ErrorHandler::eErrFunction,  129 : : pErr->getErrorMsg(  130 : : ErrorHandler::eErrFunction,  131 : : "SFTableAccessor::getSegmentForLocationPriority"));  132 : : }  133 : 8708 : return pSegPointer;  134 : : }  135 : :   136 : : Segment::Pointer  137 : : SFTableAccessor::getSegmentForPatternCode(  138 : : const LongPosition & inPosition, uint32\_t inSFLayerNo,  139 : 1536 : uint8\_t inPatternCode ) const {  140 [ - + ]: 1536 : if( inPatternCode == 0 || inPatternCode > 5 )  141 : : {  142 : 0 : return NULL;  143 : : }  144 : 1536 : Segment::Pointer pSegPointer=NULL;  145 : 1536 : try{  146 : 1536 : pSegPointer = calculateSegment( inPosition, inSFLayerNo, inPatternCode, false );  147 : : }  148 [ # # ]: 0 : catch( DPL::Error \*pChild )  149 : : {  150 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  151 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  152 : : ErrorHandler::eErrFunction,  153 : : pErr->getErrorMsg(  154 : : ErrorHandler::eErrFunction,  155 : : "SFTableAccessor::getSegmentForPatternCode"));  156 : : }  157 : 1536 : return pSegPointer;  158 : : }  159 : :   160 : : Segment::Pointer  161 : : SFTableAccessor::calculateSegment(  162 : : const LongPosition & inPosition, uint32\_t inSFLayerNo,  163 : 10244 : uint8\_t inPatternCode, bool inIsLocation ) const throw (DPL::Error \*) {  164 : 10244 : SubField::Pointer sf = getSubField( inPosition, inSFLayerNo );  165 [ - + ]: 10244 : if( NULL == sf )  166 : : {  167 : 0 : return NULL;  168 : : }  169 : 10244 : Segment::Pointer segments = 0;  170 [ + + ]: 10244 : if( inIsLocation )  171 : : {  172 : 8708 : segments = sf->getSegmentsForLocationPriority();  173 : : }  174 : : else  175 : : {  176 : 1536 : segments = sf->getSegmentsForShape(inPatternCode);  177 : : }  178 : 10244 : int64\_t posXwrtSF = inPosition.getX() - sf->getXCoordinate();  179 : 10244 : int64\_t posYwrtSF = inPosition.getY() - sf->getYCoordinate();  180 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  181 : 10244 : = mParams->getStartupMessage();  182 : 10244 : uint32\_t segWidth = pStartMsg->iSFSizeX / pStartMsg->usSegmentCountX;  183 : 10244 : uint32\_t segHeight = pStartMsg->iSFSizeY / pStartMsg->usSegmentCountY;  184 : 10244 : int64\_t idx = posXwrtSF / segWidth;  185 : 10244 : int64\_t idy = posYwrtSF / segHeight;  186 [ + - ][ - + ]: 10244 : if( idx > pStartMsg->usSegmentCountX  187 : : || idy > pStartMsg->usSegmentCountY )  188 : : {  189 : 0 : return NULL;  190 : : }  191 : 10244 : idx = (idx == pStartMsg->usSegmentCountX ) ? idx - 1: idx;  192 : 10244 : idy = (idy == pStartMsg->usSegmentCountY ) ? idy - 1: idy;  193 : 10244 : size\_t offset = 0L;  194 [ + - ][ + + ]: 10244 : switch( pStartMsg->usTFLocationOrder )  [ + - ]  195 : : {  196 : : case(eTForderUwdUwdFwd) :  197 : : case(eTForderUwdDwdFwd) :  198 : : case(eTForderDwdDwdFwd) :  199 : : case(eTForderDwdUwdFwd) :  200 : : case(eTForderUwdUwdBwd) :  201 : : case(eTForderUwdDwdBwd) :  202 : : case(eTForderDwdUwdBwd) :  203 : : case(eTForderDwdDwdBwd) :  204 : : {  205 : : //Column major  206 : 8708 : offset = idx \* pStartMsg->usSegmentCountY + idy;  207 : 8708 : break;  208 : : }  209 : : case(eTForderFwdFwdUwd) :  210 : : case(eTForderFwdBwdUwd) :  211 : : case(eTForderBwdBwdUwd) :  212 : : case(eTForderBwdFwdUwd) :  213 : : case(eTForderFwdFwdDwd) :  214 : : case(eTForderFwdBwdDwd) :  215 : : case(eTForderBwdBwdDwd) :  216 : : case(eTForderBwdFwdDwd) :  217 : : {  218 : : //Row major  219 : 1536 : offset = idy \* pStartMsg->usSegmentCountX + idx;  220 : 10244 : break;  221 : : }  222 : : }  223 : : //TF TFBuffers do not exist, create buffer  224 : : Segment::Pointer segment  225 : 10244 : = segments + offset;  226 [ + - ]: 10244 : if( segment )  227 : : {  228 : 10244 : TFBuffer::Pointer buf = segment->getTFs();   229 [ + + ]: 10244 : if( 0 == buf )  230 : : {  231 : 4768 : boost::mutex::scoped\_lock l(mTFBufCreationLock);  232 : 4768 : if( 0 == buf )  233 : : {  234 : : bool keepTF = pStartMsg->usProcessingMode == eProcessNormal  235 : 4768 : || pStartMsg->usProcessingMode == eProcessArbitrary;  236 : 4768 : try{  237 : 4768 : segment->createBuffer( keepTF );  238 : : }  239 [ # # ]: 0 : catch( DPL::Error \*pChild )  240 : : {  241 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  242 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  243 : : ErrorHandler::eErrFunction,  244 : : pErr->getErrorMsg(  245 : : ErrorHandler::eErrFunction,  246 : : "SFTableAccessor::calculateSegment"));  247 : : }  248 : :   249 : : //In normal mode .. tF goes back to a buffer to be re-used..  250 : : //TODO::OPTIMIZE FOR TFLESS MODE?  251 : : }  252 : : }  253 : : }  254 [ # # ]: 10244 : return segment;  255 : : }  256 : :   257 : 1215 : } // namespace nft |

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