# Pieces versus Directions (/Pieces+versus+Directions)



Table of Contents



**Home * Board Representation * Bitboards * Pieces versus Directions**

Despite the generation of moves to one target square by pieces from different directions as mentioned Square Attacked By, there are two approaches to generate moves in an origin centric manner to determine a set of distinct target squares. The classical approach, also applied in Mailbox array representation, is to generate attacks and moves piece by piece. Alternately there are set-wise approaches, to determine distinct direction attacks for sets of multiple pieces and pawns. Sliding pieces rely on dumb7fill, kogge-stone algorithm or fill by subtraction.

## Piece by Piece

In general there are lookup techniques to determine attack- or move-target sets for each individual pawn or piece, indexed by bitscanned square and for the sliders by occupancy. Each piece-wise attack covers all their inherited ray- or knight directions. This classical approach traverses all pieces and generates attack sets and move sets by following techniques:

- Pawn Push
- Attacks of a Single Pawn
- Knight Attacks by Lookup
- King Attacks by Lookup
- Single Sliding Piece Attacks

As usual, captures are determined by intersection of attacks with opponent pieces or en passant target, while all other attacks or push targets of empty squares are quiet moves.

```
for all pieces {
  generateMovetargets(piece);
  for all capture targets
      generate captures (move or capture list)
  for all empty square targets
      generate quite moves (move list)
}
```

## Directions

The direction-wise approach relies on following set-wise techniques to determine move-targets. As usual, captures are determined by intersection of attacks with opponent pieces or en passant target, while all other attacks or push targets of empty squares are quiet moves.

- Pushes set-wise
- Pawn Attacks set-wise
- Multiple Knight Attacks
- King Attacks by Calculation with Direction-Steps
- Multiple Sliding Pieces

For move generation purpose one may store target-sets of different piece types per direction. Thus, 16 bitboards as a kind of unsorted move list. Each target square in each ray-direction set has an unique one-to-one relation to it's source square.

```
for all 16 ray- and knight directions {
  generateMovetargets(dirtargets[dir], dir);
}
```

Likely this loop is unrolled - at least for the ray-attacks - where pawns, kings, bishops, rooks or queens may involved. See the proposal of DirGolem for a branch-less direction-wise generation of legal moves.

An incremental finite state machine - move-getter may scan and reset the target squares per direction. For distant source squares of sliding pieces, one may intersect the ray-wise masks per direction and target-square with the occupancy - to use either bitscan reverse or forward for the positive or negative directions - similar to the classical approach, except one don't needs the if (blocker) - condition, which is always true. However, one may delay determining the source square and moving piece until make move time, and to encode moves uniquely with target square (six bits) and

direction (4 bits).



Alternatively one may process a ray in one run, to bitscan the most farthest destination square from one distinct direction, i.e. bitscan reverse of positive ray attacks, and to loop over all intermediate targets by adding offsets until the source square is arrived. The below loop considers that move-target bits may already reset due to staged move generation, i.e. hash move or killer moves may already tried. Therefor the extra condition with *testAndResetBit* [1].

```
while ( northMoveTargets )
{
    targetSquare = bitScanReverse( northMoveTargets );
    sourceSquare = bitScanReverse( rayAttacks[sout][targetSquare] & occupied);
    storeMoveOrCapture( sourceSquare, targetSquare);
    resetBit (northMoveTargets, targetSquare);
    targetSquare -= 8;
    while ( targetSquare > sourceSquare) {
       if ( testAndResetBit (northMoveTargets, targetSquare) )
          storeMove( sourceSquare, targetSquare);
       targetSquare -= 8;
    }
}
```

## Hybrid

One may combine both approaches with usual move lists. For instance direction-wise attacks for all pawns, piece-wise attacks for none-pawns. The target set-of pawn-captures should be considered early, since this are winning or at least equal captures.

## See also

- Bitboard Serialization
- BitScan
- Classical Approach of Sliding Piece Attacks
- DirGolem
- Move Generation
- Square Attacked By

## External Links

- Miles Davis - Call It Anything , Isle of Wight Festival , August 29, 1970, YouTube Video
  Miles Davis, Gary Bartz , Chick Corea, Keith Jarrett, Dave Holland, Jack DeJohnette, Airto Moreira [2] [3]
  Directions (Joe Zawinul) 0:00 (7:30), Bitches Brew 7:30 (10:09), It's About That Time 17:39 (6:17),
  Sanctuary (Wayne Shorter) 23:56 (1:10), Spanish Key 25:06 (8:15), The Theme 33:21 (2:10)


## References

1. ^ _bittestandreset, _bittestandreset64
2. ^ Does anyone know the name of the instrument Airto Moreira uses during the Isle of Wight Festival (1970)? | Yahoo! Answers
3. ^ Cuíca from Wikipedia

## What links here?

| Page | Date Edited |
|---|---|
| Attack and Defend Maps | Nov 5, 2016 |
| Bitboard Serialization | Dec 24, 2014 |

| Page | Date Edited |
|---|---|
| Bitboards | Nov 14, 2017 |
| Classical Approach | Jan 28, 2018 |
| Direction | Oct 6, 2016 |
| DirGolem | Jun 5, 2016 |
| Dumb7Fill | May 27, 2016 |
| Encoding Moves | Mar 27, 2016 |
| Fill by Subtraction | Mar 5, 2010 |
| Kogge-Stone Algorithm | Sep 17, 2016 |
| Move Generation | Jan 29, 2018 |
| Move List | Jul 19, 2017 |
| Origin Square | Oct 10, 2016 |
| Pieces versus Directions | Oct 6, 2016 |
| Target Square | Oct 26, 2017 |

**Up one Level**