Initialization

In[ ]:=
```
<< KnotTheory`
```

```
Loading KnotTheory` version of February 2, 2020, 10:53:45.2097.
Read more at http://katlas.org/wiki/KnotTheory.
```

Formulas

Alexander's Original Definition (crossings × faces)

In[ ]:=
```
AlexanderOriginal[K_] := T ↦ Apart@Module[
    {XingsByTurns, bends, faces, p, A, is, poly},
    XingsByTurns =
     PD[K] /. x : X[i_, j_, k_, l_] :→
       If[PositiveQ[x], X+[-i, j, k, -l], X-[-j, k, l, -i]];
    bends = Times @@ XingsByTurns /.
      _[X][a_, b_, c_, d_] :→ p[a,-d] p[b,-a] p[c,-b] p[d,-c];
    faces = bends //. p[x__,y_] p[y_,z__] :→ p[x,y,z];
    A = Table[0, Length@XingsByTurns, Length@faces];
    Do[is = Position[faces, #][[1, 1]] & /@ List @@ XingsByTurns[[j]];
     A[[{j}, is]] = If[Head[XingsByTurns[[j]]] === X+,
       (-T 1 -1 T), (1 -1 T -T)],
     {j, Length@XingsByTurns}];
    poly = Det@A[[All, Delete[Range[Length@faces],
          {Position[faces, #][[1, 1]]} & /@ {1, -1}]]];
    (poly /. T → 1) poly / T^((Exponent[poly,T,Max]+Exponent[poly,T,Min])/2)
    ];
```

Bar-Natan & Dancso (crossings × crossings)

```
In[∘]:= AlexanderBND[K_] := T ↦ Apart@Module[{Xings, n = Length@PD[K], Sp,
          I = IdentityMatrix[Length@PD[K]], outgoingStrands, A, σ, d},
         outgoingStrands[x_] := List @@ x⟦{If[PositiveQ[x], 2, 4], 3}⟧;
         (*Cutting along 2n→1 strand*)
         Xings = PD[K] /. x : X[a_, b_, c_, d_] /;
             MemberQ[outgoingStrands[x], 1] ⧴ Replace[x, 1 → 2 n + 1, {-1}];
         Sp[i_] := Delete[
           Range[Sequence @@ Sort@outgoingStrands[Xings⟦i⟧]],
           -1];
         A = Table[If[DisjointQ[List @@ Xings⟦j, {1, 3}⟧, Sp[i]], 0, 1],
            {i, n}, {j, n}] - I;
         σ = Table[If[PositiveQ[Xings⟦i⟧], 1, -1], {i, n}];
         d = Table[If[Xings⟦i, 1⟧ ≥ Xings⟦i, 2⟧, 1, -1], {i, n}];
         Det[I + A.(I - DiagonalMatrix[T^(-σ*d)])]];
```

Long Position ((edges + 1) × (edges + 1))

```
In[∘]:= Rot[pd_PD] := Module[{n, xs, x, rots, Xp, Xm, front = {1}, k},
       n = Length@pd; rots = Table[0, {2 n}];
       xs = Cases[pd, x_X ⧴ { Xp[x⟦4⟧, x⟦1⟧]  PositiveQ@x
                              Xm[x⟦2⟧, x⟦1⟧]  True      ];
       For[k = 1, k ≤ 2 n, ++k,
        If[FreeQ[front, -k],
          front = Flatten@Replace[front, k → (xs /. {
                 Xp[k, l_] | Xm[l_, k] ⧴ {l + 1, k + 1, -l},
                 Xp[l_, k] | Xm[k, l_] ⧴ (++rots⟦l⟧; {-l, k + 1, l + 1}),
                 _Xp + _Xm ⧴ {}
                }), {1}],
          Cases[front, k | -k] /. {k, -k} ⧴ --rots⟦k⟧;
         ];];
       {xs /. {Xp[i_, j_] ⧴ {+1, i, j}, Xm[i_, j_] ⧴ {-1, i, j}}, rots}];
   Rot[K_] := Rot[PD[K]]
```

```
In[◦]:= AlexanderLongPosition[K_] := T ↦ Apart@Module[{A, s, is, φ, w},
    A = IdentityMatrix[2 * Length@PD[K] + 1];
    s[x_] := If[PositiveQ@x, 1, -1];
    Do[is = If[PositiveQ@x, List @@ x[[{4, 1}]], List @@ x[[{2, 1}]]];
     A[[is, # + 1 & /@ is]] = ( -T^s[x]  T^s[x] - 1
                                  0         -1     ),
     {x, List @@ PD[K]}];
    φ = Total[Rot[K][[2]]]; w = Total[s /@ List @@ PD[K]];
    T^(-(φ+w)/2) Det[A]]
```

Arc Presentation

```
In[◦]:= MinesweeperMatrix[ap_] := Module[{l = Length[ap], currentRow, c1, c2, s},
   currentRow = Table[0, {l}];
   Table[{c1, c2} = Sort[ap[[k]]];
    s = Sign[{-1, 1}.ap[[k]]];
    Do[currentRow[[c]] += s, {c, c1, c2 - 1}];
    currentRow,
    {k, l}]]

In[◦]:= AlexanderViaArcPresentation[K_] :=
  T ↦ Apart@Module[{ap = ArcPresentation[K], M, fixPower, indices},
    M = MinesweeperMatrix[ap];
    indices[c_, i_] := Sequence[
      If[i === 1, {i}, {i, i - 1}], If[c === 1, {c}, {c, c - 1}]];
    fixPower = -1/8 Sum[
       Total[M[[indices[ap[[i, 1]], i]]], 2] + Total[M[[indices[ap[[i, 2]], i]]], 2],
       {i, Length@ap}];
    Signature[List @@ ap[[All, 2]]] T^fixPower (T^(-1/2) - T^(1/2))^(1-Length[ap]) Det[T^M]]
```

Burau Representation (Braids)

```
In[∘]:= AlexanderViaBurau[K_] := T ↦ Apart@Module[{m, n, br, reducedBurau,
      poly, δ, σ},
     {n, br} = List @@ BR[K];
     δ /: δ_{i_,j_} := If[i == j, 1, 0];
     σ_{i_}[ℰ_] := { ℰ /. v_1 :→ -T * v_1 + v_2                      i === 1
                   { ℰ /. v_i :→ T * v_{i-1} - T * v_i + v_{i+1}     1 < i < n - 1;
                   { ℰ /. v_{n-1} :→ T * v_{n-2} - T * v_{n-1}       i === n - 1
     m_{i_} := σ_i[Table[v_j, {j, n - 1}]] /. v_{j_} :→ Table[δ_{j,k}, {k, n - 1}];
     reducedBurau = Dot @@ br /. i_Integer :→ If[i > 0, m_i, m_{Abs[i]} // Inverse];
     poly = (1 - T)/(1 - T^n) Det[IdentityMatrix[n - 1] - reducedBurau] // Apart;
     (poly /. T → 1) poly / T^Mean[{Exponent[poly,T,Max],Exponent[poly,T,Min]}]
    ];
```

Seifert Matrix (via Braids)

```
In[∘]:= AlexanderViaSeifertAsBraids[K_] := T ↦ Apart@Module[{br, h, V},
    br = BR[K]〚2〛;
    h = With[{absBR = Abs /@ br},
      Table[
       Append[Position[absBR〚i + 1 ;;〛, absBR〚i〛] + i, {0}]〚1, 1〛, {i, Length@absBR - 1}]];
    V = Table[Which[
         h〚i〛 h〚j〛 === 0, Nothing,
         h〚Min[i, j]〛 > h〚Max[i, j]〛, 0,
         h〚Min[i, j]〛 < Max[i, j], 0,

         i === j, -(Sign[br〚i〛] + Sign[br〚h〚i〛〛])/2,
         h〚i〛 === j ∧ br〚j〛 < 0, -1,
         h〚j〛 === i ∧ br〚i〛 > 0, 1,
         i < j ∧ Abs[br〚i〛] - Abs[br〚j〛] === -1, 1,
         j < i ∧ Abs[br〚j〛] - Abs[br〚i〛] === 1, -1,
         True, 0],
       {i, Length@h}, {j, Length@h}] /. {} → Nothing;
    T^(-Length[V]/2) Det[V - T * Transpose[V]]]
```

Testing

Testing on the Rolfsen Table (excluding the unknot).

In[∘]:=
```
Sum[Equal @@ {Alexander[K][T], AlexanderOriginal[K][T], AlexanderBND[K][T],
    AlexanderLongPosition[K][T], AlexanderViaArcPresentation[K][T],
    AlexanderViaBurau[K][T], AlexanderViaSeifertAsBraids[K][T]},
  {K, AllKnots[{3, 10}]}]
```

KnotTheory: Loading precomputed data in PD4Knots`.

KnotTheory: MorseLink was added to KnotTheory` by Siddarth Sankaran
at the University of Toronto in the summer of 2005.

KnotTheory: The minimum braids representing the knots with up to 10 crossings were provided by Thomas Gittings. See arXiv:math.GT/0401051.

Out[∘]=
```
249 True
```

We can also include prime knots with 11 crossings in our test, though this computation takes several minutes.

In[∘]:=
```
Sum[Equal @@ {Alexander[K][T], AlexanderOriginal[K][T], AlexanderBND[K][T],
    AlexanderLongPosition[K][T], AlexanderViaArcPresentation[K][T],
    AlexanderViaBurau[K][T], AlexanderViaSeifertAsBraids[K][T]},
  {K, AllKnots[{3, 11}]}]
```

KnotTheory: Loading precomputed data in DTCode4KnotsTo11`.

KnotTheory: The GaussCode to PD conversion was written by Siddarth Sankaran at the University of Toronto in the summer of 2005.

KnotTheory: Vogel's algorithm was implemented by Dan Carney in the summer of 2005 at the University of Toronto.

Out[∘]=
```
801 True
```

Conjecture Testing

Alexander's Original Definition (crossings × faces)

In[∘]:=
```
AlexanderOriginalConjectured[K_] := T ↦ Apart@Module[
   {XingsByTurns, bends, faces, p, A, is, poly,
    σ, d, Xings = List @@ PD[K]},
   XingsByTurns =
    Xings /. x : X[i_, j_, k_, l_] :>
      If[PositiveQ[x], Subscript[X, +][-i, j, k, -l], Subscript[X, -][-j, k, l, -i]];
   bends = Times @@ XingsByTurns /.
     _[X][a_, b_, c_, d_] :> Subscript[p, a, -d] Subscript[p, b, -a] Subscript[p, c, -b] Subscript[p, d, -c];
   faces = bends //. Subscript[p, x__, y_] Subscript[p, y_, z__] :> Subscript[p, x, y, z];
   A = Table[0, Length@XingsByTurns, Length@faces];
   Do[is = Position[faces, #][[1, 1]] & /@ List @@ XingsByTurns[[j]];
    A[[{j}, is]] = If[Head[XingsByTurns[[j]]] === Subscript[X, +],
      (-T 1 -1 T), (1 -1 T -T)],
    {j, Length@XingsByTurns}];
   poly = Det@A[[All, Delete[Range[Length@faces],
        {Position[faces, #][[1, 1]]} & /@ {1, -1}]]];
   σ = Table[If[PositiveQ[x], 1, -1], {x, Xings}];
   d = Table[If[x[[1]] ≥ Max[List @@ x[[{2, 4}]]], 1, -1], {x, Xings}];
   ((poly /. T → 1) poly) / T^Count[σ*d, -1]
   ];
```

In[∘]:=
```
Sum[AlexanderOriginalConjectured[K][T] === Alexander[K][T], {K, AllKnots[{3, 11}]}]
```

Out[∘]=
```
801 True
```

Burau Representation (Braids)

In[∘]:=
```
AlexanderViaBurauConjectured[K_] := T ↦ Apart@Module[{b, m, n, br, reducedBurau,
    poly, δ, σ},
   {n, br} = List @@ BR[K];
   δ /: Subscript[δ, i_, j_] := If[i == j, 1, 0];
   Subscript[σ, i_][ℰ_] := Piecewise[{
      {ℰ /. Subscript[v, 1] :> -T*Subscript[v, 1] + Subscript[v, 2], i === 1},
      {ℰ /. Subscript[v, i] :> T*Subscript[v, i-1] - T*Subscript[v, i] + Subscript[v, i+1], 1 < i < n-1},
      {ℰ /. Subscript[v, n-1] :> T*Subscript[v, n-2] - T*Subscript[v, n-1], i === n-1}}];
   Subscript[m, i_] := Subscript[σ, i][Table[Subscript[v, j], {j, n-1}]] /. Subscript[v, j_] :> Table[Subscript[δ, j, k], {k, n-1}];
   reducedBurau = Dot @@ br /. i_Integer :> If[i > 0, Subscript[m, i], Subscript[m, Abs[i]] // Inverse];
   1/T^((1+Total[Sign/@br]-n)/2) (1-T)/(1-T^n) Det[IdentityMatrix[n-1] - reducedBurau]];
```

In[ ]:=
```
Sum[Alexander[K][T] === AlexanderViaBurauConjectured[K][T], {K, AllKnots[{3, 11}]}]
```
Out[ ]=
```
801 True
```