# B-Splines and the de Boor Algorithm

**Ron Goldman**
**Department of Computer Science**
**Rice University**

## Primary Properties of the Blossom

*i. Symmetry*

$$p(u_1,\ldots,u_n) = p(u_{\sigma(1)},\ldots,u_{\sigma(n)}) \quad \text{for any permutation } \sigma \text{ of } \{1,\ldots,n\}.$$

*ii. Multiaffine*

$$p(u_1,\ldots,(1-\alpha)u_k + \alpha w_k,\ldots,u_n) = (1-\alpha)p(u_1,\ldots,u_k,\ldots,u_n) + \alpha p(u_1,\ldots,w_k,\ldots,u_n)$$

*iii. Diagonal*

$$P(t) = p(\underbrace{t,\ldots,t}_{n})$$

*iv. Differentiation*

$$P^{(k)}(t) = \frac{n!}{(n-k)!}\,p(\underbrace{t,\ldots,t}_{n-k},\underbrace{\delta,\ldots,\delta}_{k})$$

*v. Dual Functional*

$$P_k = p(\underbrace{a,\ldots,a}_{n-k},\underbrace{b,\ldots,b}_{k})$$

# <u>The Multiaffine Property</u>

$$\begin{tikzcd}
& p(a,a,t) & \\
p(a,a,a) \arrow[ur, "\frac{b-t}{b-a}"] & & p(a,a,b) \arrow[ul, "\frac{t-a}{b-a}"']
\end{tikzcd}$$

Normalized

$$\begin{tikzcd}
& aat & \\
aaa \arrow[ur, "b-t"] & & aab \arrow[ul, "t-a"']
\end{tikzcd}$$

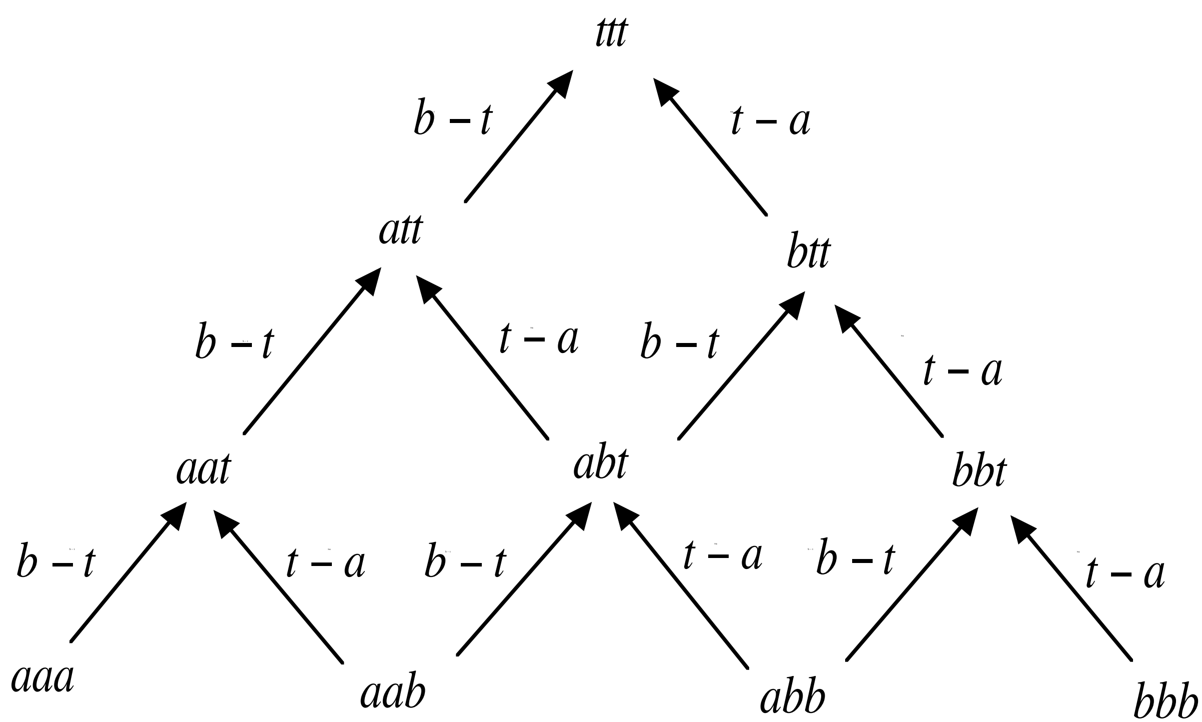
Unnormalized

$$t = \frac{b-t}{b-a}a + \frac{t-a}{b-a}b$$

$$p(a,a,t) = p\left(a,a,\frac{b-t}{b-a}a + \frac{t-a}{b-a}b\right) = \frac{b-t}{b-a}p(a,a,a) + \frac{t-a}{b-a}p(a,a,b)$$

$$aat = \frac{b-t}{b-a}aaa + \frac{t-a}{b-a}aab$$

# <u>de Casteljau's Algorithm</u>




$$\underbrace{a\cdots a}_{i}\underbrace{b\cdots b}_{j}\underbrace{t\cdots t}_{k} = p(\underbrace{a,\ldots,a}_{i},\underbrace{b,\ldots,b}_{j},\underbrace{t,\ldots,t}_{k})$$

$$t = \frac{b-t}{b-a}a + \frac{t-a}{b-a}b$$

**Multiaffine Property**

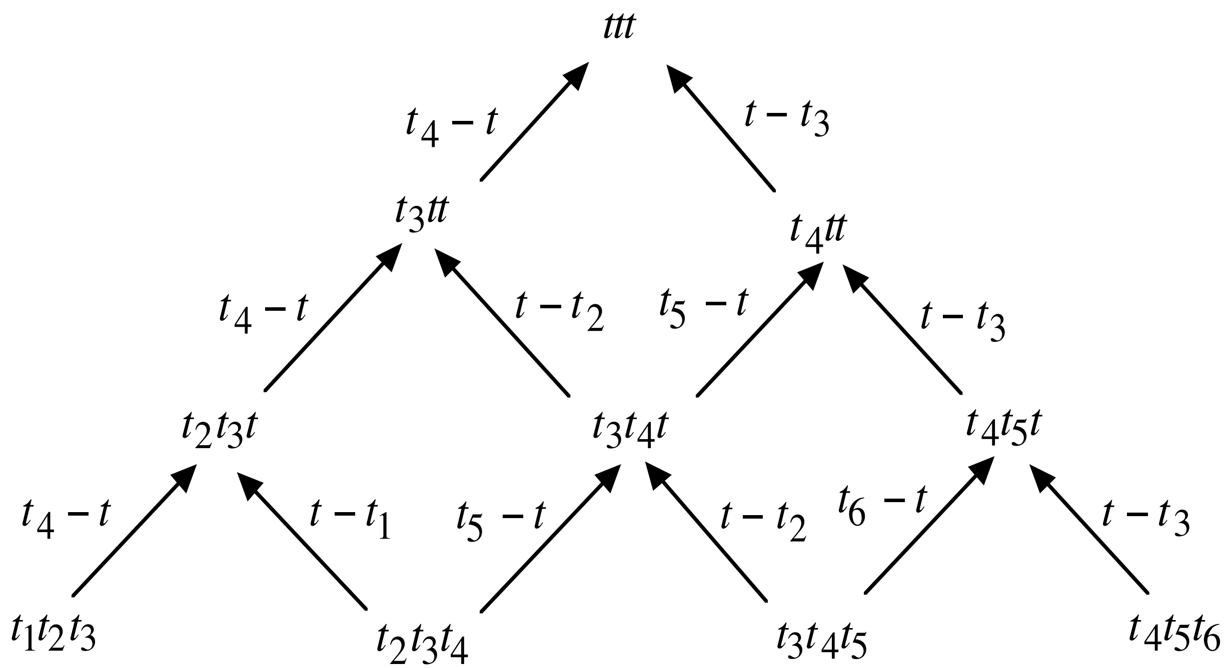
$$\begin{tikzcd}[column sep=small] & t_2 t_3 t & \\ t_1 t_2 t_3 \arrow[ur, "t_4 - t"] & & t_2 t_3 t_4 \arrow[ul, "t - t_1"'] \end{tikzcd}$$

$$t = \frac{t_4 - t}{t_4 - t_1} t_1 + \frac{t - t_1}{t_4 - t_1} t_4$$

$$p(t_2, t_3, t) = \frac{t_4 - t}{t_4 - t_1} p(t_1, t_2, t_3) + \frac{t - t_1}{t_4 - t_1} p(t_2, t_3, t_4)$$

$$t_2 t_3 t = \frac{t_4 - t}{t_4 - t_1} t_1 t_2 t_3 + \frac{t - t_1}{t_4 - t_1} t_2 t_3 t_4$$

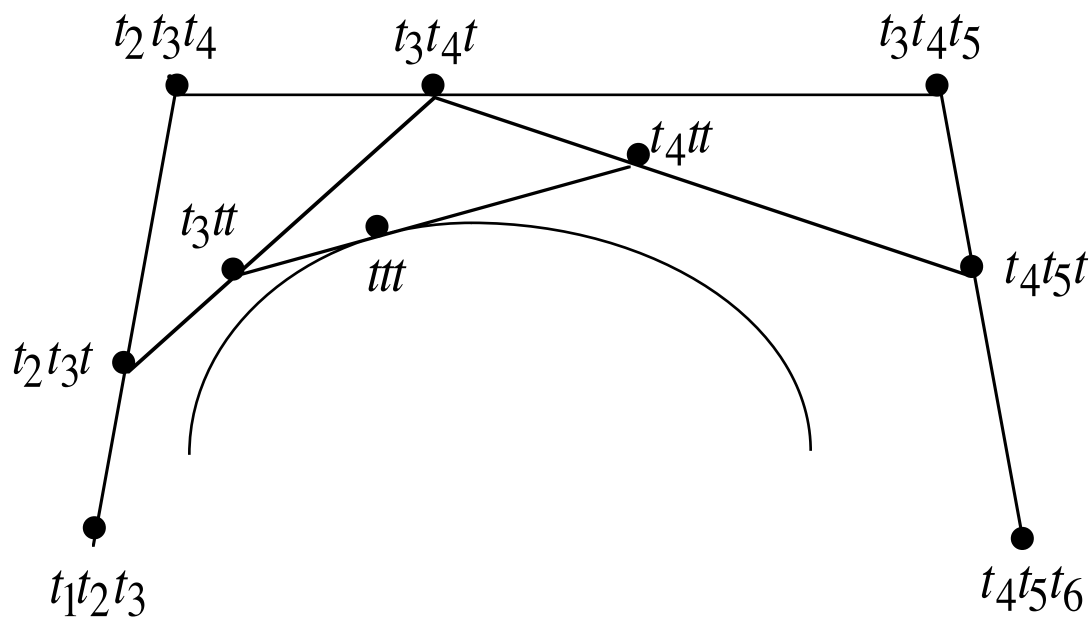
# <u>**De Boor Algorithm**</u>




One Polynomial Segment: $t_3 \le t \le t_4$

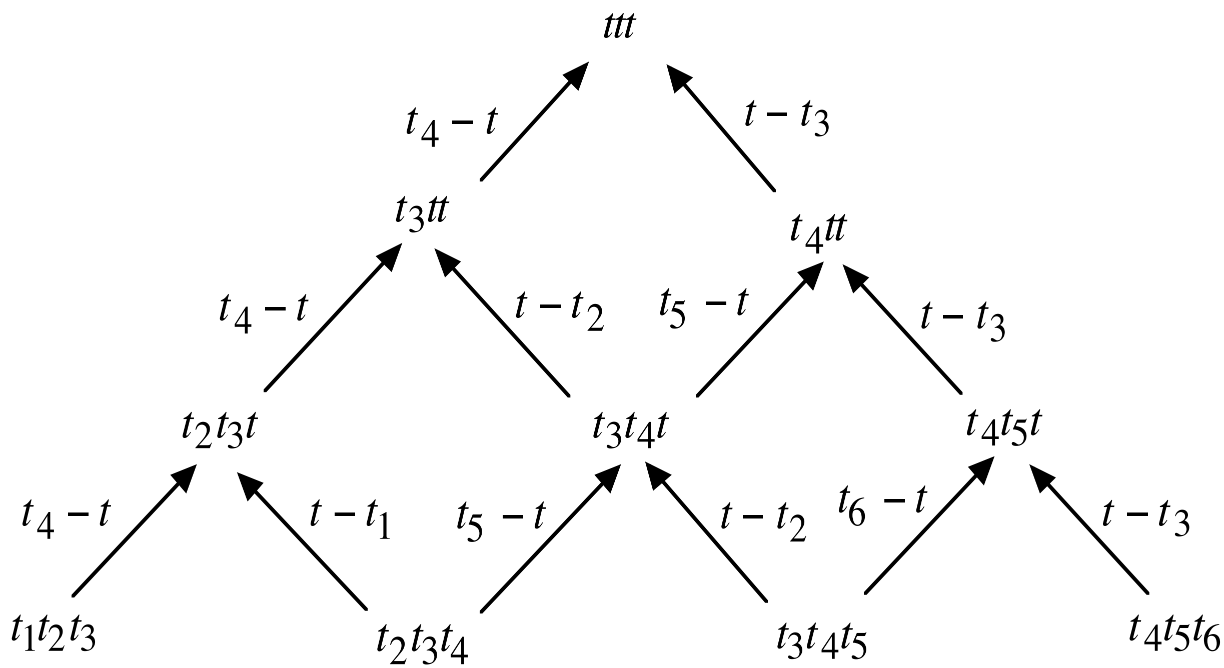
Knots: $t_1 \le t_2 \le t_3 < t_4 \le t \le t_5 \le t_6$

# B-Spline Curve




One Polynomial Segment: $t_3 \leq t \leq t_4$

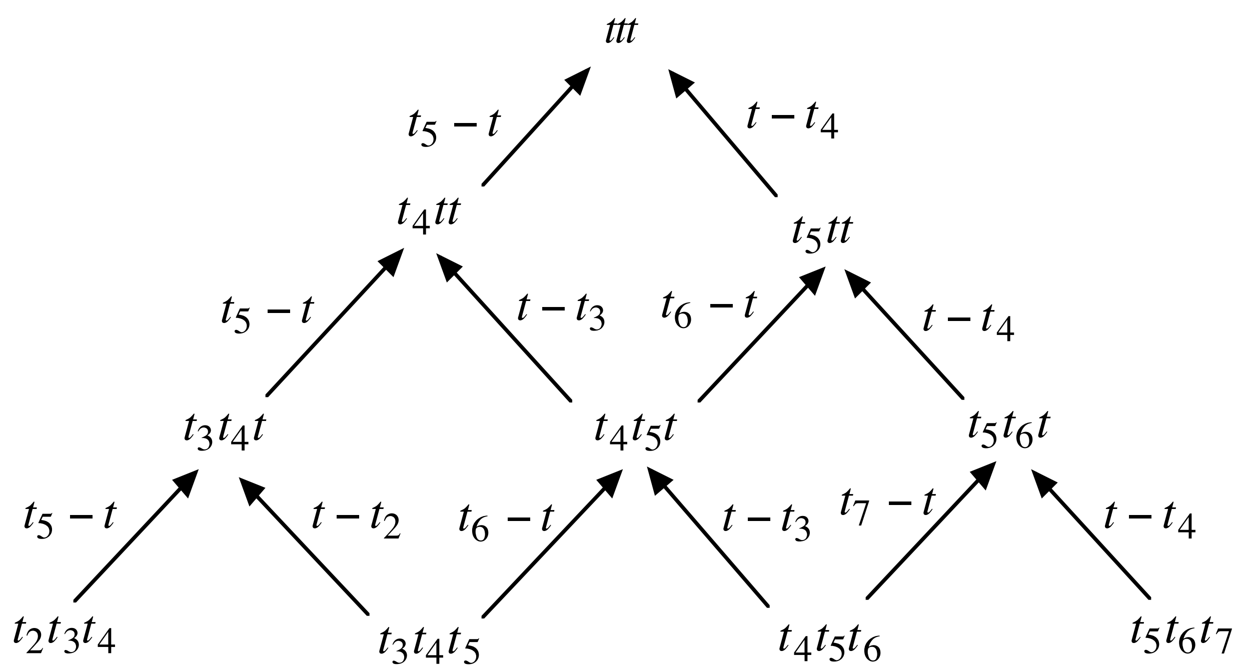
# <u>De Boor Algorithm</u>




One Polynomial Segment: $t_3 \leq t \leq t_4$

Knots: $t_1 \leq t_2 \leq t_3 < t_4 \leq t \leq t_5 \leq t_6$

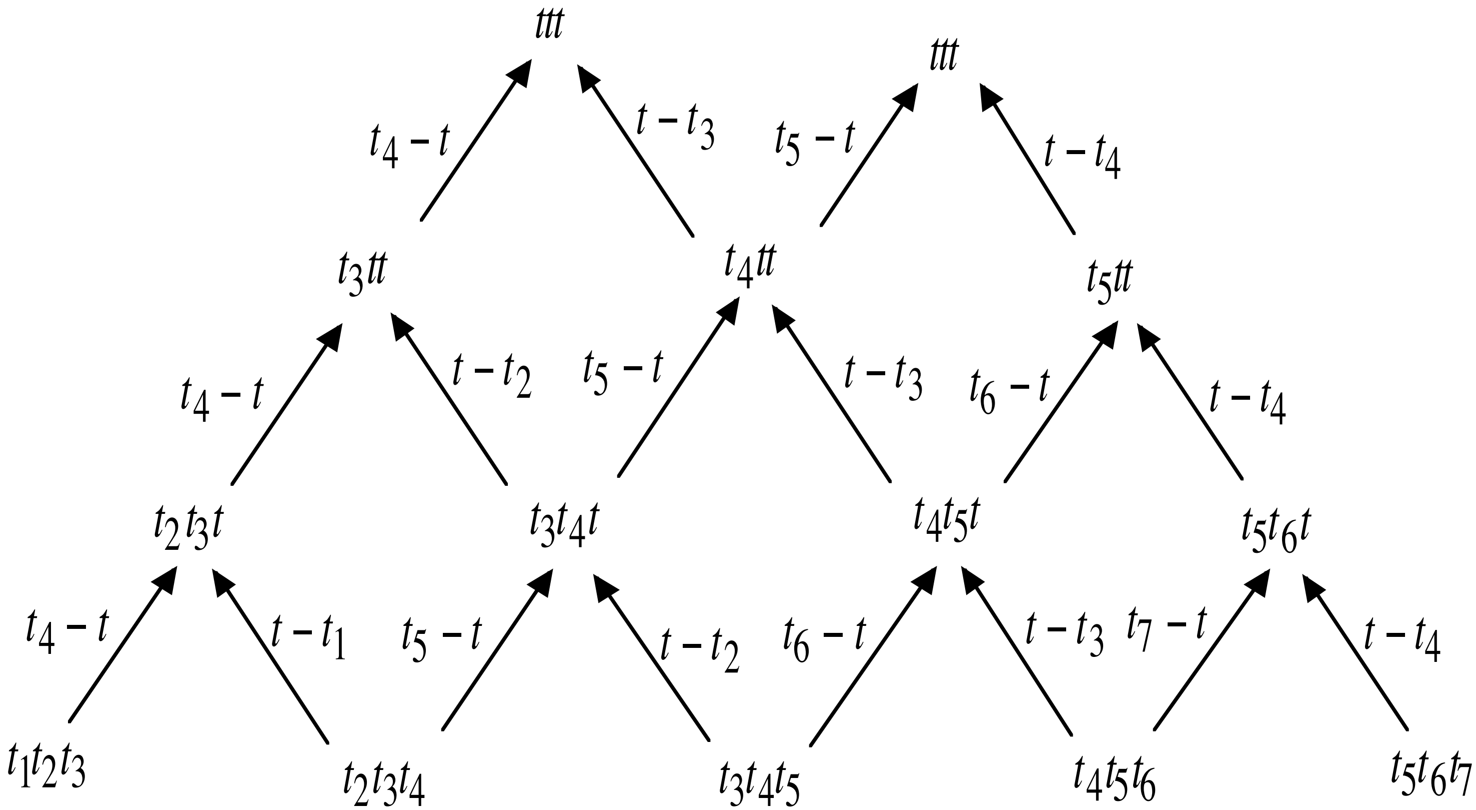
# De Boor Algorithm




One Polynomial Segment: $t_4 \le t \le t_5$

Knots: $t_2 \le t_3 \le t_4 < t_5 \le t \le t_6 \le t_7$

# De Boor Algorithm




Two Polynomial Segments: $t_3 \le t \le t_4$ and $t_4 \le t \le t_5$

Knots: $t_1 \le t_2 \le t_3 < t_4 < t_5 \le t \le t_6 \le t_7$

# <u>**Dual Functional Property**</u>

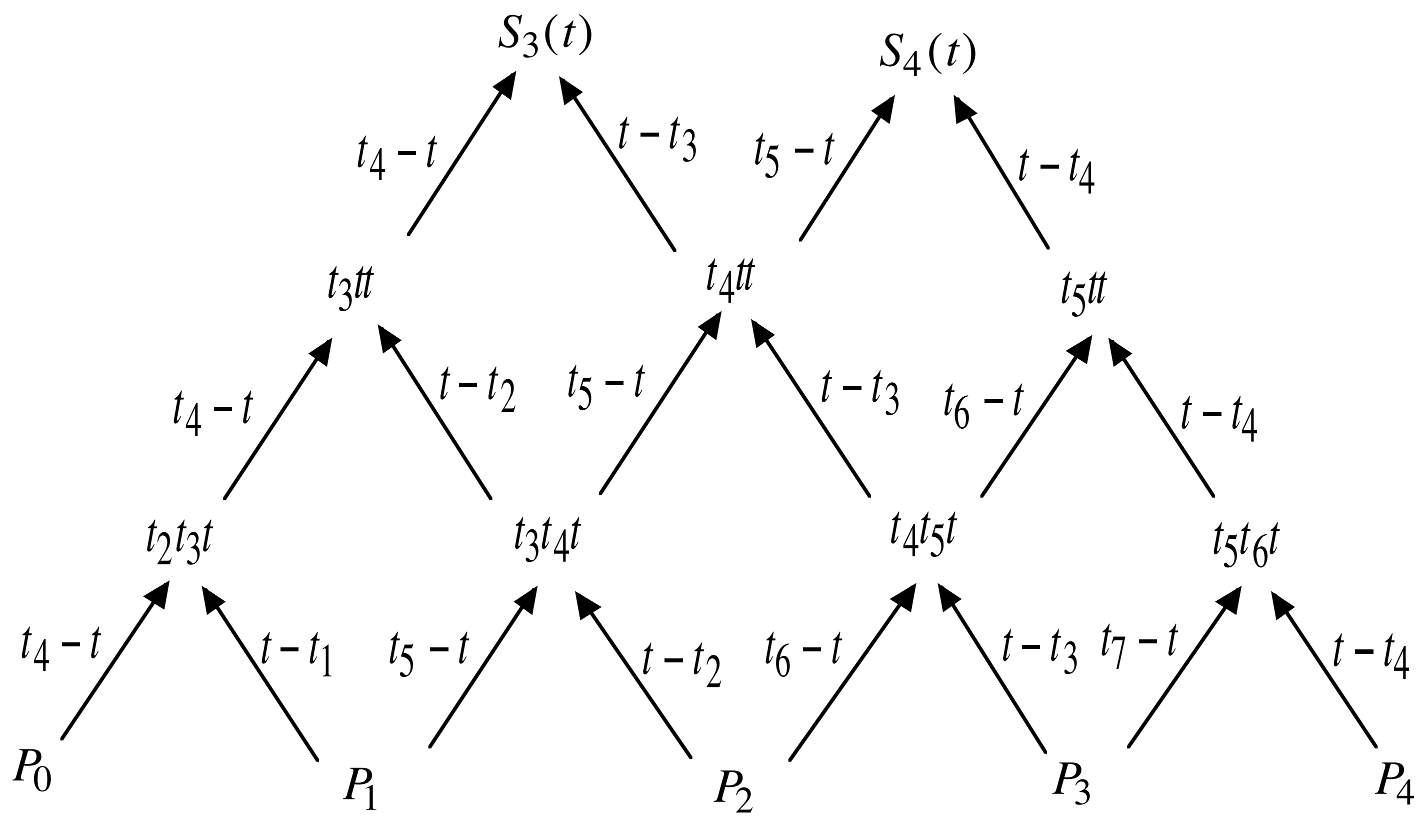
*B-Spline Curves*

- $T = \{t_0, \ldots, t_L\} =$ knot vector
- $S(t) =$ B-spline curve
- $s(t_{k+1}, \ldots, t_{k+n}) = P_k = k^{th}$ B-Spline Control Point

*Bezier Curves*

- $T = \{\underbrace{a, \ldots, a}_{n}, \underbrace{b, \ldots, b}_{n}\} =$ knot vector
- $P(t) =$ Bezier curve
- $p(\underbrace{a, \ldots, a}_{n-k}, \underbrace{b, \ldots, b}_{k}) = P_k = k^{th}$ Bezier Control Point

## De Boor Algorithm




Two Polynomial Segments: $t_3 \le t \le t_4$ and $t_4 \le t \le t_5$

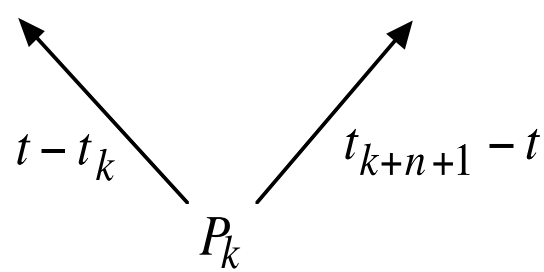
Knots: $t_1 \le t_2 \le t_3 < t_4 < t_5 \le t_6 \le t_7$

# <u>Advantages of the de Boor Algorithm</u>

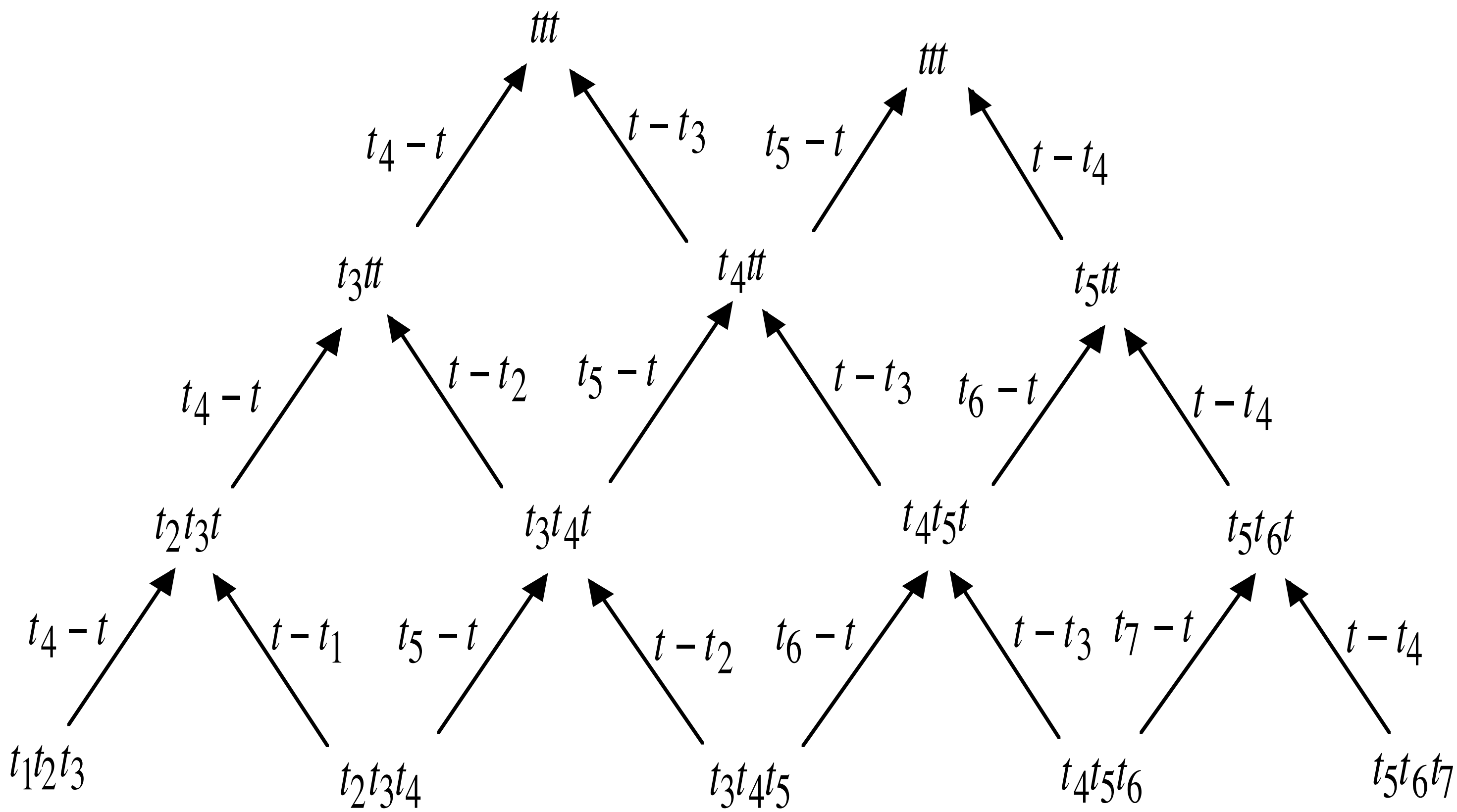
1. *Numerically Stable*
   - Uses the Given Data Directly
   - Computes Only Convex Combinations

2. *Flexible*
   - Automatic Smoothness Between Polynomial Segments
   - No Special Positioning of the Control Points

3. *Simple Structure*
   - In-Out Property
   - Basic Step of the de Boor Algorithm



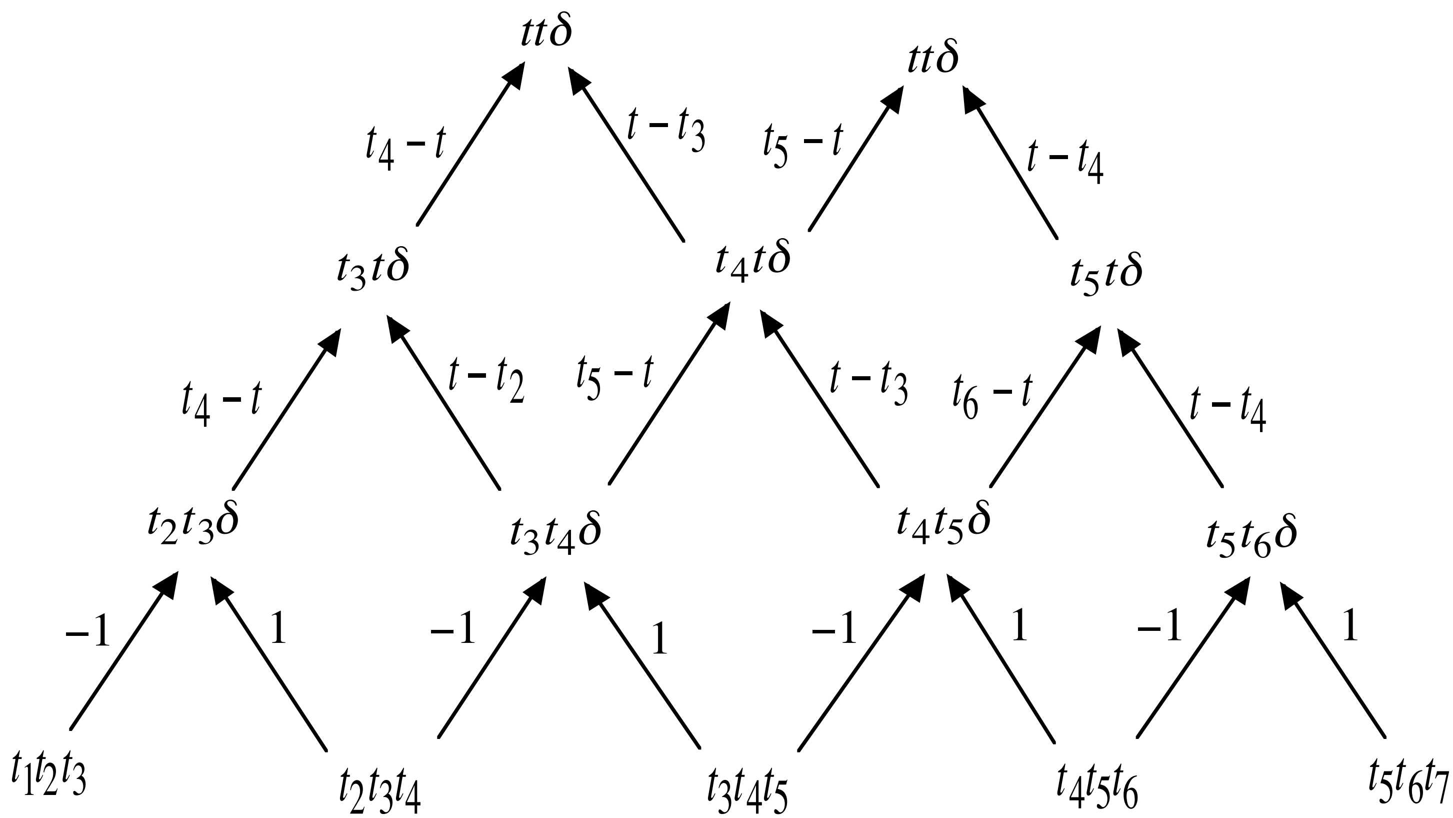
# De Boor Algorithm -- Evaluation




$C^0$ *Continuity*

$ttt = t_4tt$ at $\mathrm{t} = \mathrm{t}_4$

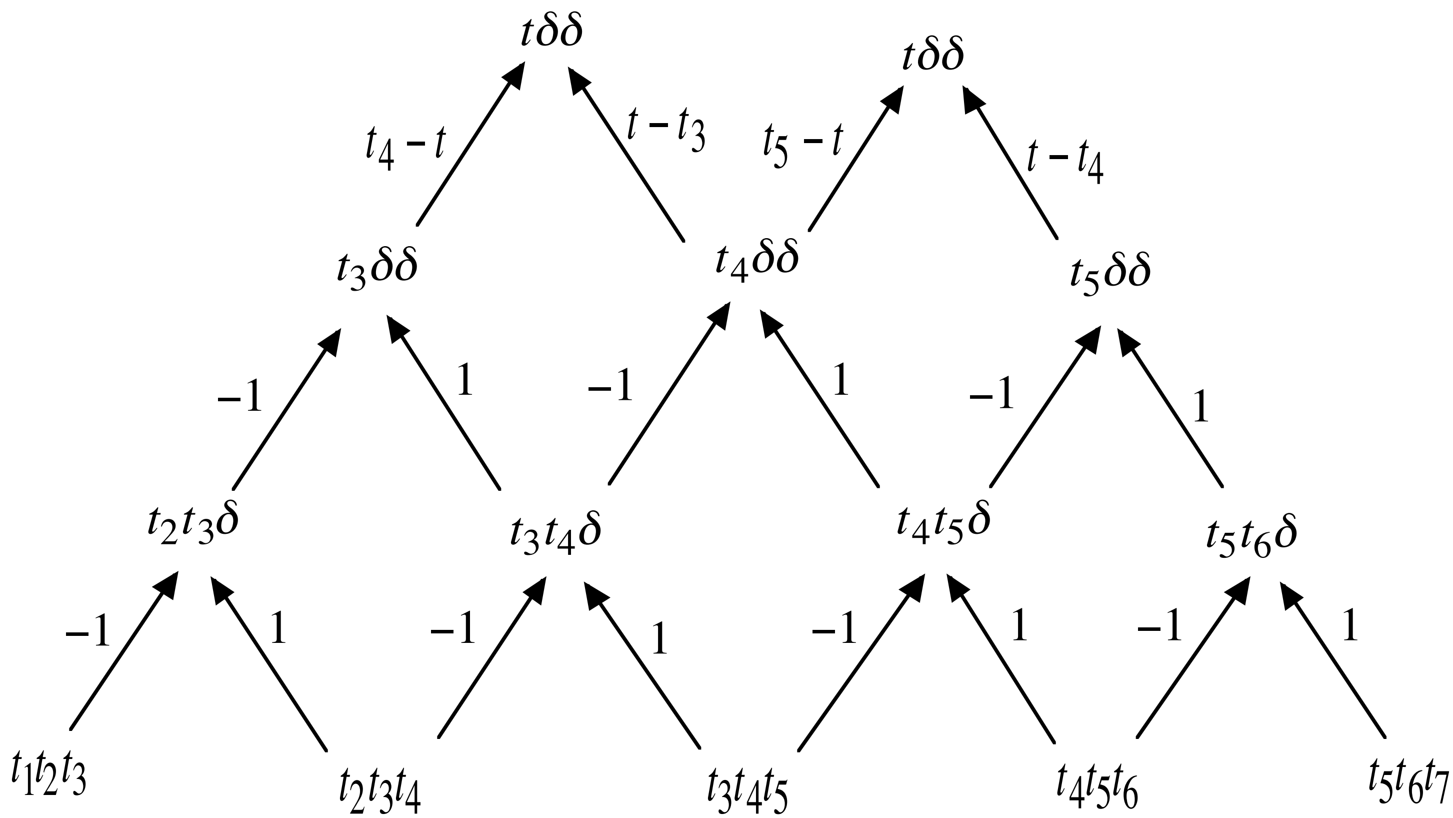
# De Boor Algorithm -- First Derivative




$C^1$ *Continuity*

$tt\delta = t_4 t\delta$ at $\mathrm{t} = \mathrm{t}_4$

# De Boor Algorithm -- Second Derivative



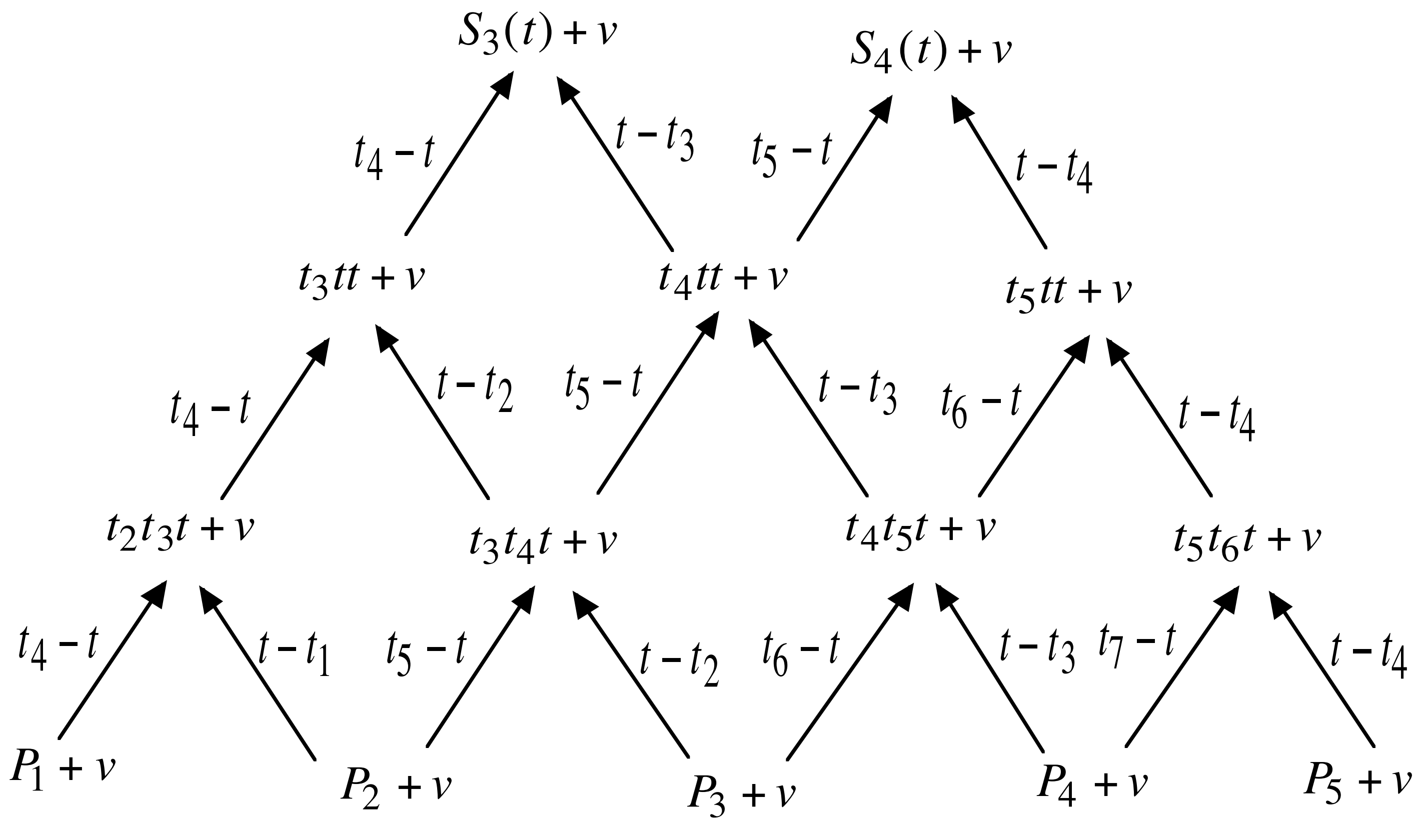

*$C^2$ Continuity*

$t\delta\delta = t_4\delta\delta$ at $t = t_4$

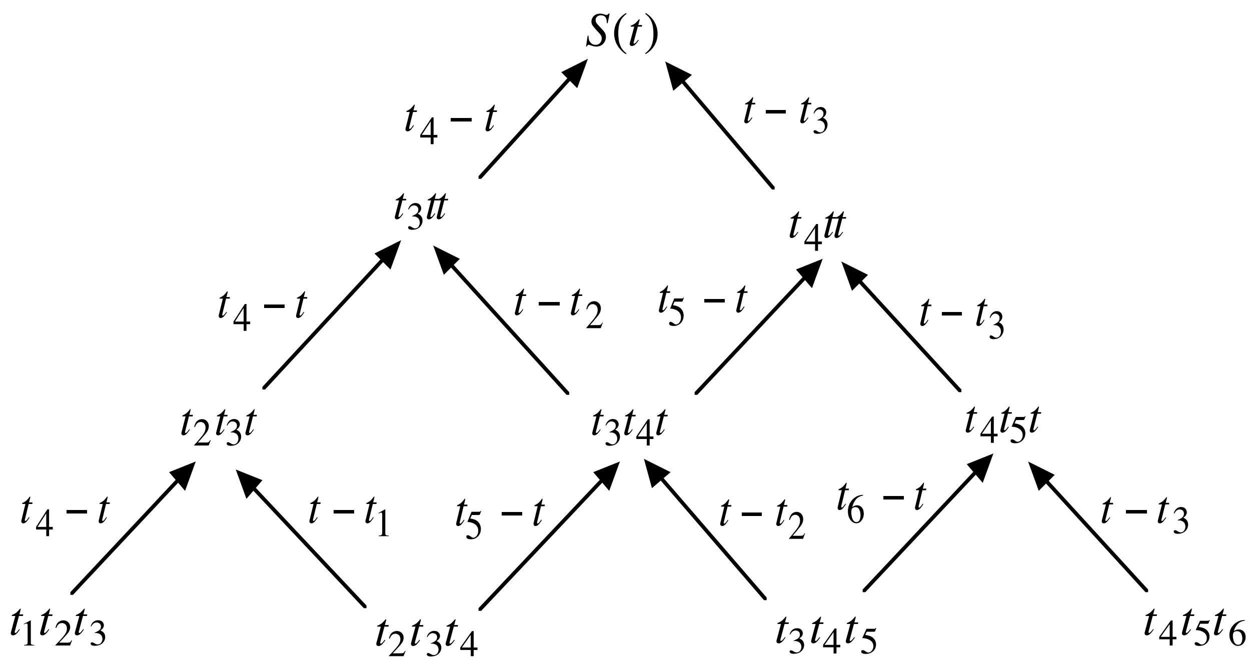
## Elementary Properties of B-Spline Curves

1. Piecewise polynomial
2. Continuity of order $C^{n-\mu}$ at knots of multiplicity $\mu$ on curves of degree $n$
3. Translation invariance
4. Local convex hull property
5. Interpolation at knots where the multiplicity $\mu$ is equal to the degree $n$
6. Local control

# **Translation Invariance**



# <u>Interpolation</u>

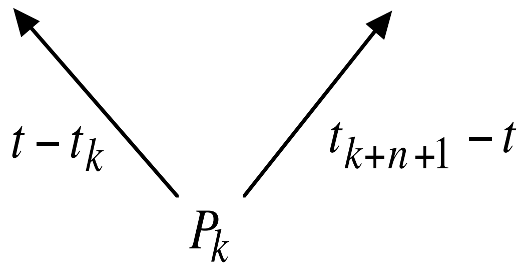



*Interpolation at knots of multiplicity* $n = 3$

$t_1 = t_2 = t_3 \Rightarrow S(t_3) = t_3t_3t_3 = t_1t_2t_3$

# <u>Local Control</u>

*Basic Step of the de Boor Algorithm*



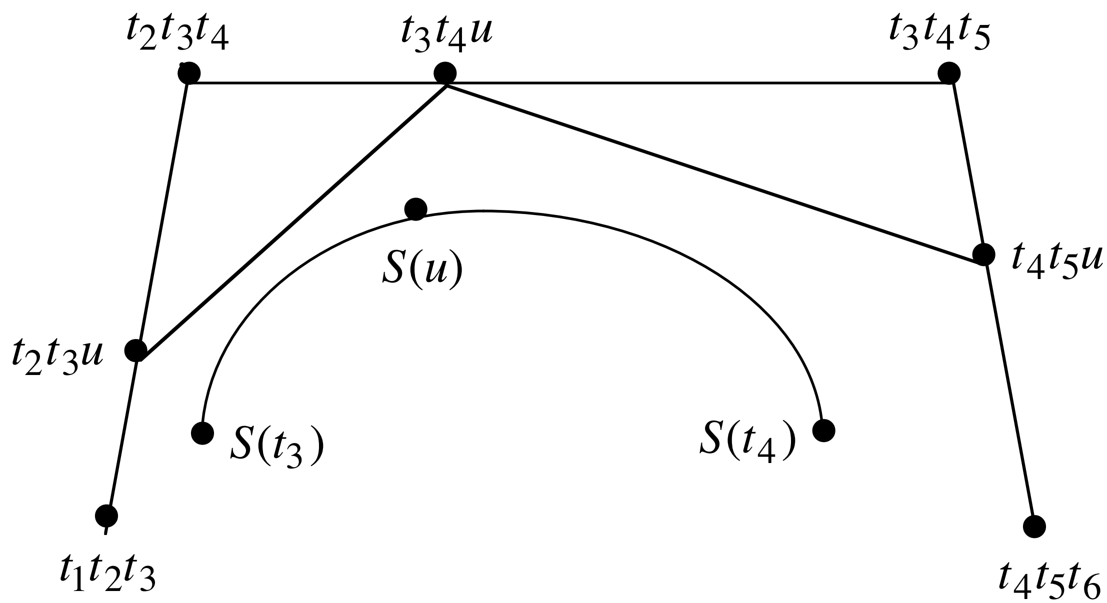

*kth Segment of a Degree n B-spline Curve*

- lies over the parameter interval $[t_k, t_{k+1}]$
- has $n+1$ control points -- $P_{k-n}, \ldots, P_k$
- depends on $2n$ knots -- $t_{k-n+1}, \ldots, t_{k+n}$

*Local Control*

- $P_k$ influences only the $n+1$ curve segments over the interval $[t_k, t_{k+n+1}]$
- $t_k$ influences solely the $2n$ curve segments over the interval $[t_{k-n}, t_{k+n}]$

## Knot Insertion




*Inserting the knot u between the knots $t_3$ and $t_4$,*
*changes the control polygon, but does not alter the curve.*

# Knot Insertion

*Algorithms*

- Boehm's Algorithm
- Oslo Algorithm
- Lane Riesenfeld Algorithm

*Applications*

- Conversion to Piecewise Bezier Form
- Rendering
- Intersection
- Variation Diminishing Property

# Boehm's Knot Insertion Algorithm

*Insert One Knot at a Time*

Replace $\ldots, t_{k-1}, t_k, t_{k+1}, t_{k+2}, \ldots$ with $\ldots, t_{k-1}, t_k, u, t_{k+1}, t_{k+2}, \ldots$

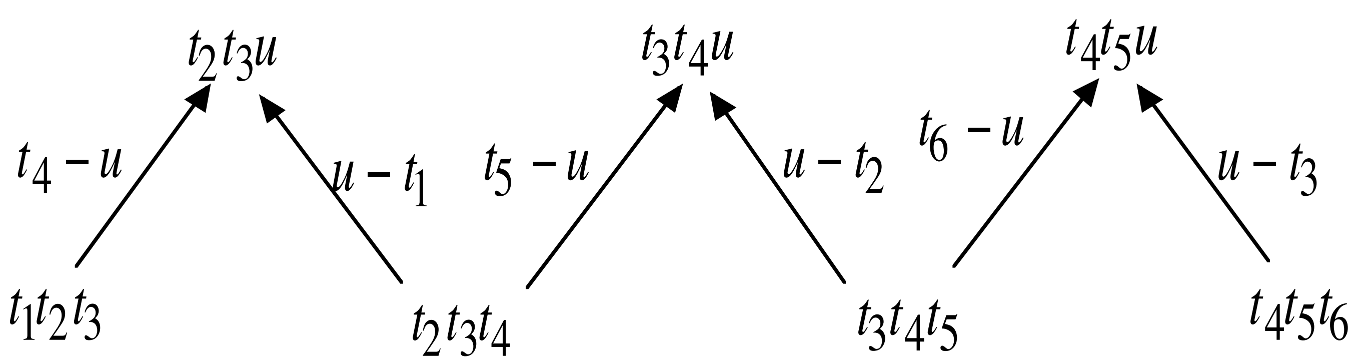
*Old Control Points*

$$p(t_{k-2}, t_{k-1}, t_k),\ p(t_{k-1}, t_k, t_{k+1}),\ p(t_k, t_{k+1}, t_{k+2}),\ p(t_{k+1}, t_{k+2}, t_{k+3})$$

*New Control Points*

$$p(t_{k-2}, t_{k-1}, t_k),\ p(t_{k-1}, t_k, u),\ p(t_k, u, t_{k+1}),\ p(u, t_{k+1}, t_{k+2}),\ p(t_{k+1}, t_{k+2}, t_{k+3})$$

New Control Points




Original Control Points

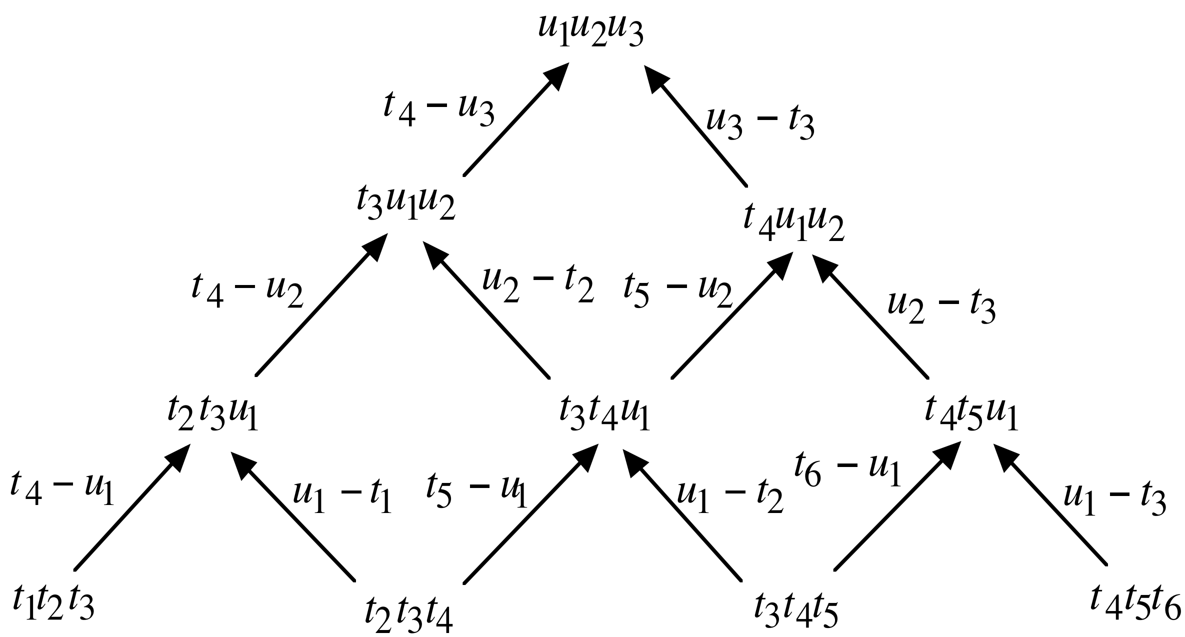
# Oslo Knot Insertion Algorithm -- Blossoming the de Boor Algorithm

*Inserts Many Knots Simultaneously*

Replace $\ldots, t_{k-1}, t_k, t_{k+1}, t_{k+2}, \ldots$ with $\ldots, t_{k-1}, t_k, u_1, \ldots, u_p, t_{k+1}, t_{k+2}, \ldots$

*Compute One New Control Point at a Time*

| Old Control Points | New Control Points |
|---|---|
| $\{p(t_{k+1}, t_{k+2}, t_{k+3})\}$ | $\{p(u_{i+1}, u_{i+2}, u_{i+3})\}$ |



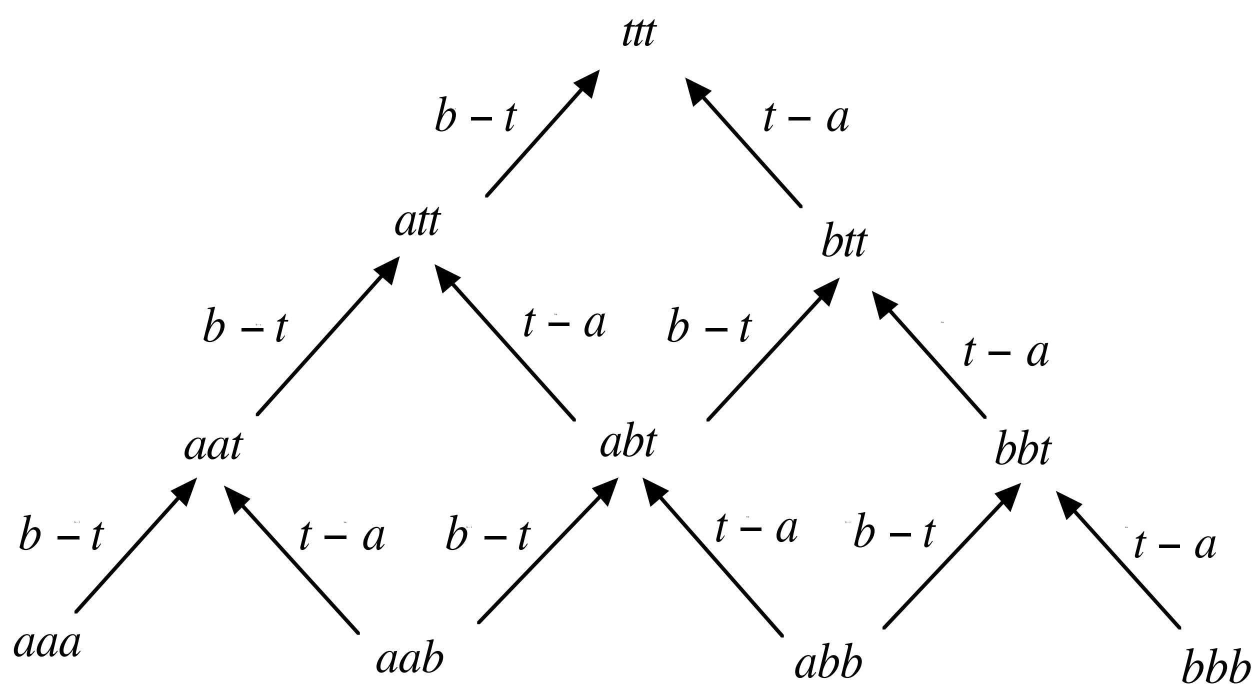
## Special Cases

de Boor Evaluation Algorithm

de Casteljau Subdivision Algorithm

Conversion to Piecewise Bezier Form

# De Casteljau Subdivision Algorithm



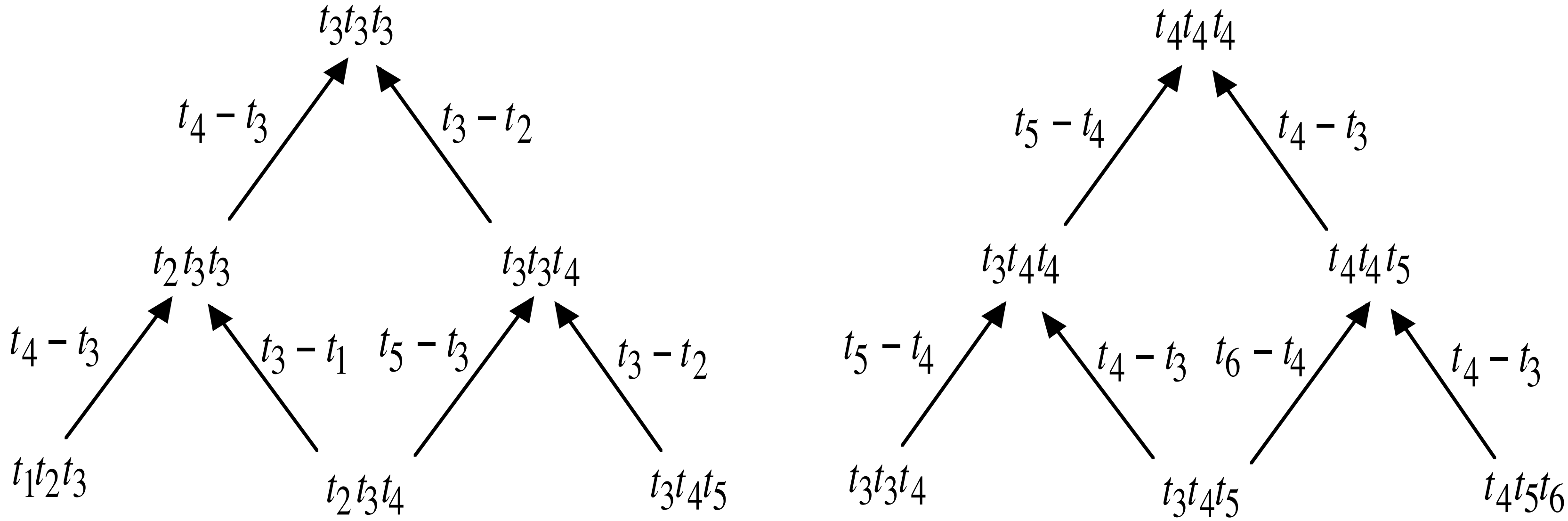

*Inserting a Multiple Knot at t*

# Conversion to Piecewise Bezier Form

*Inserts Multiple Knots*

Replace $\ldots, t_{k-1}, t_k, t_{k+1}, t_{k+2}, \ldots$ with $\underbrace{t_k, \ldots, t_k}_{n}, \underbrace{t_{k+1}, \ldots, t_{k+1}}_{n}$

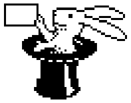
Cubic Segment: $t_1, t_k, t_2, t_3, t_4, t_5, t_6 \rightarrow t_3, t_3, t_3, t_4, t_4, t_4$



**Theorem:** *B-spline curves are variation diminishing*.

Proof:

- Knot insertion is a corner cutting procedure.
- Therefore the piecewise Bezier control polygon is variation diminishing with respect to the original B-spline control polygon.
- But each Bezier segment is variation diminishing with respect to its Bezier control polygon.
- Moreover the B-spline curve and the piecewise Bezier curve are identical.
- Hence the entire B-spline curve must be variation diminishing with respect to the original B-spline control polygon.



# **Uniform B-splines**

*Uniform B-splines*

- Evenly space knots
- Knots at the integers

*Knot Insertion Algorithms*

- Insert knots at the half integers
- Control Polygon Converges to B-spline Curve

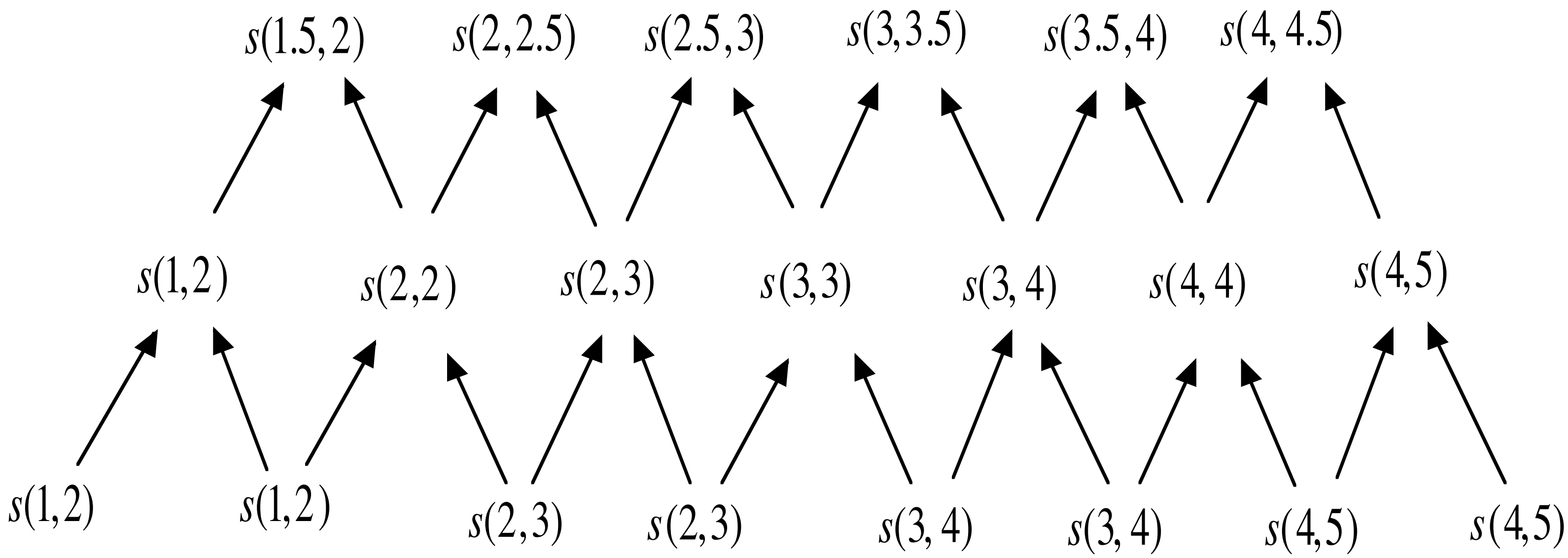
*Examples*

- Chaikin’s Algorithm and the Lane-Riesenfeld Algorithm
- Split and Average

*Applications*

- Rendering
- Intersection

# **Chaikin's Algorithm**




*Inserting Knots at the Half Integers: Quadratic Case*
*Double and Average*

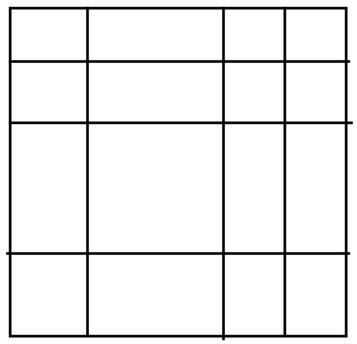
# Lane-Riesenfeld Algorithm

$$
\begin{tikzcd}[column sep=tiny]
 & & & \frac{4P_0+4P_1}{8} & & \frac{P_0+6P_1+P_2}{8} & & \frac{4P_1+4P_2}{8} & & \frac{P_1+6P_2+P_3}{8} & & \frac{4P_2+4P_3}{8} & & & \\
 & & \frac{3P_0+P_1}{4} \arrow[ur] & & \frac{P_0+3P_1}{4} \arrow[ul] \arrow[ur] & & \frac{3P_1+P_2}{4} \arrow[ul] \arrow[ur] & & \frac{P_1+3P_2}{4} \arrow[ul] \arrow[ur] & & \frac{3P_2+P_3}{4} \arrow[ul] \arrow[ur] & & \frac{P_2+3P_3}{4} \arrow[ul] & & \\
 & P_0 \arrow[ur] & & \frac{P_0+P_1}{2} \arrow[ul] \arrow[ur] & & P_1 \arrow[ul] \arrow[ur] & & \frac{P_1+P_2}{2} \arrow[ul] \arrow[ur] & & P_2 \arrow[ul] \arrow[ur] & & \frac{P_2+P_3}{2} \arrow[ul] \arrow[ur] & & P_3 \arrow[ul] & \\
P_0 \arrow[ur] & & P_0 \arrow[ul] \arrow[ur] & & P_1 \arrow[ul] \arrow[ur] & & P_1 \arrow[ul] \arrow[ur] & & P_2 \arrow[ul] \arrow[ur] & & P_2 \arrow[ul] \arrow[ur] & & P_3 \arrow[ul] \arrow[ur] & & P_3 \arrow[ul]
\end{tikzcd}
$$

*Inserting Knots at the Half Integers: Cubic Case*

*Double and Average*

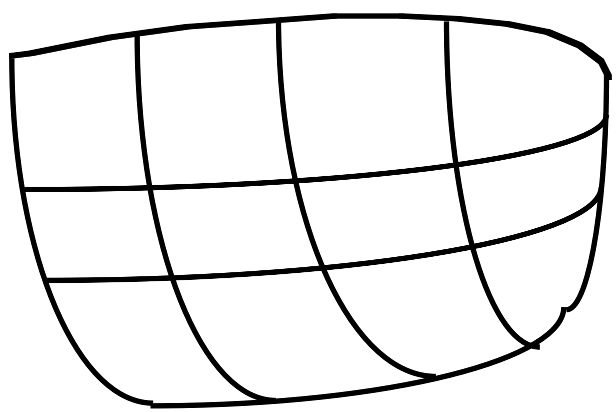
# Tensor Product B-Spline Surfaces



*Domain Rectangle: Knot Lines*

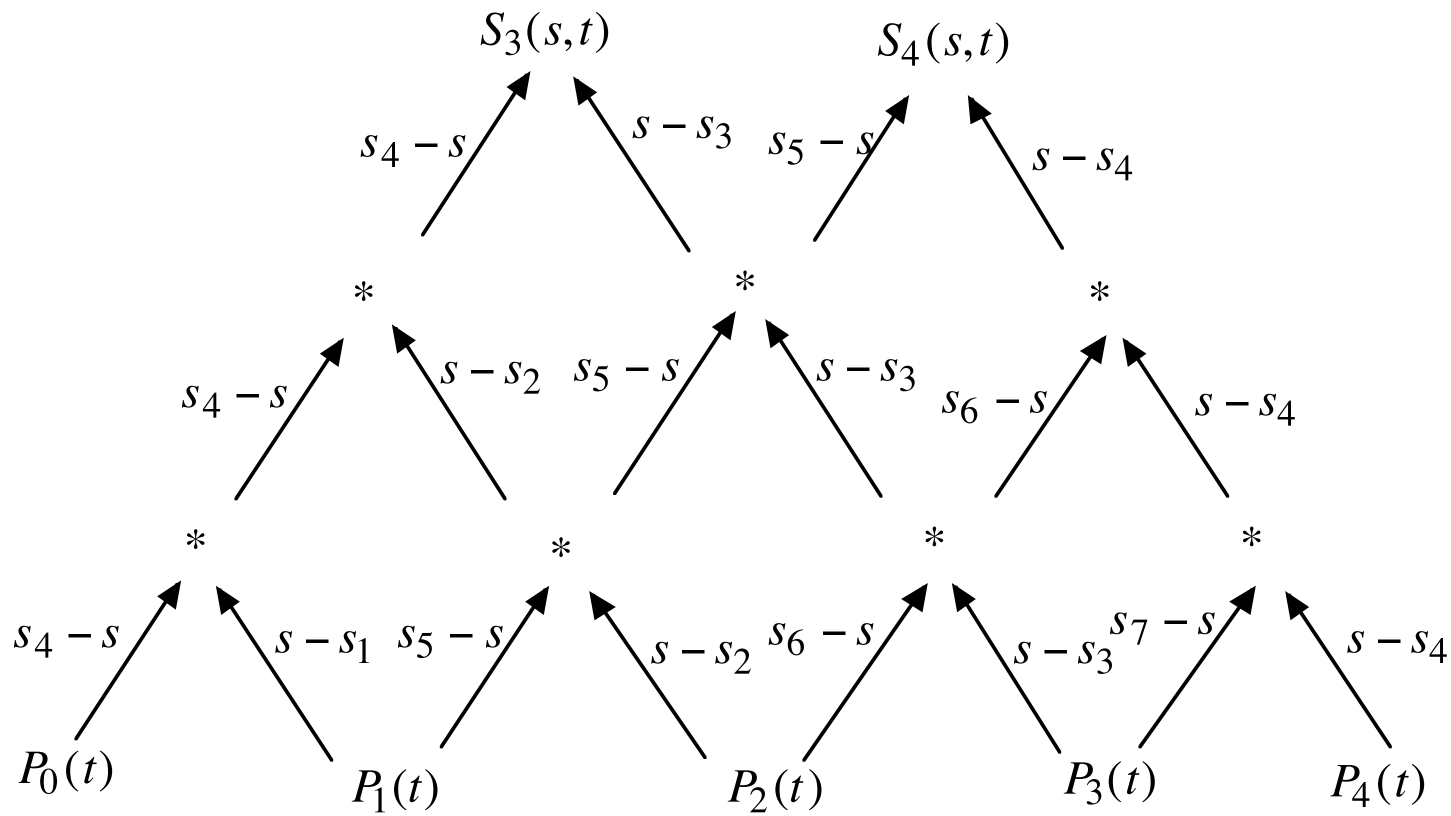
| | | | | | | |
|---|---|---|---|---|---|---|
| $P_{03}$ | $P_{13}$ | $P_{23}$ | $P_{33}$ | $P_{43}$ | $P_{53}$ | |
| $P_{02}$ | $P_{12}$ | $P_{22}$ | $P_{32}$ | $P_{42}$ | $P_{52}$ | $\bullet\bullet\bullet$ |
| $P_{01}$ | $P_{11}$ | $P_{21}$ | $P_{31}$ | $P_{41}$ | $P_{51}$ | |
| $P_{00}$ | $P_{10}$ | $P_{20}$ | $P_{30}$ | $P_{40}$ | $P_{50}$ | |

*Range: Rectangular Array of Control Points*



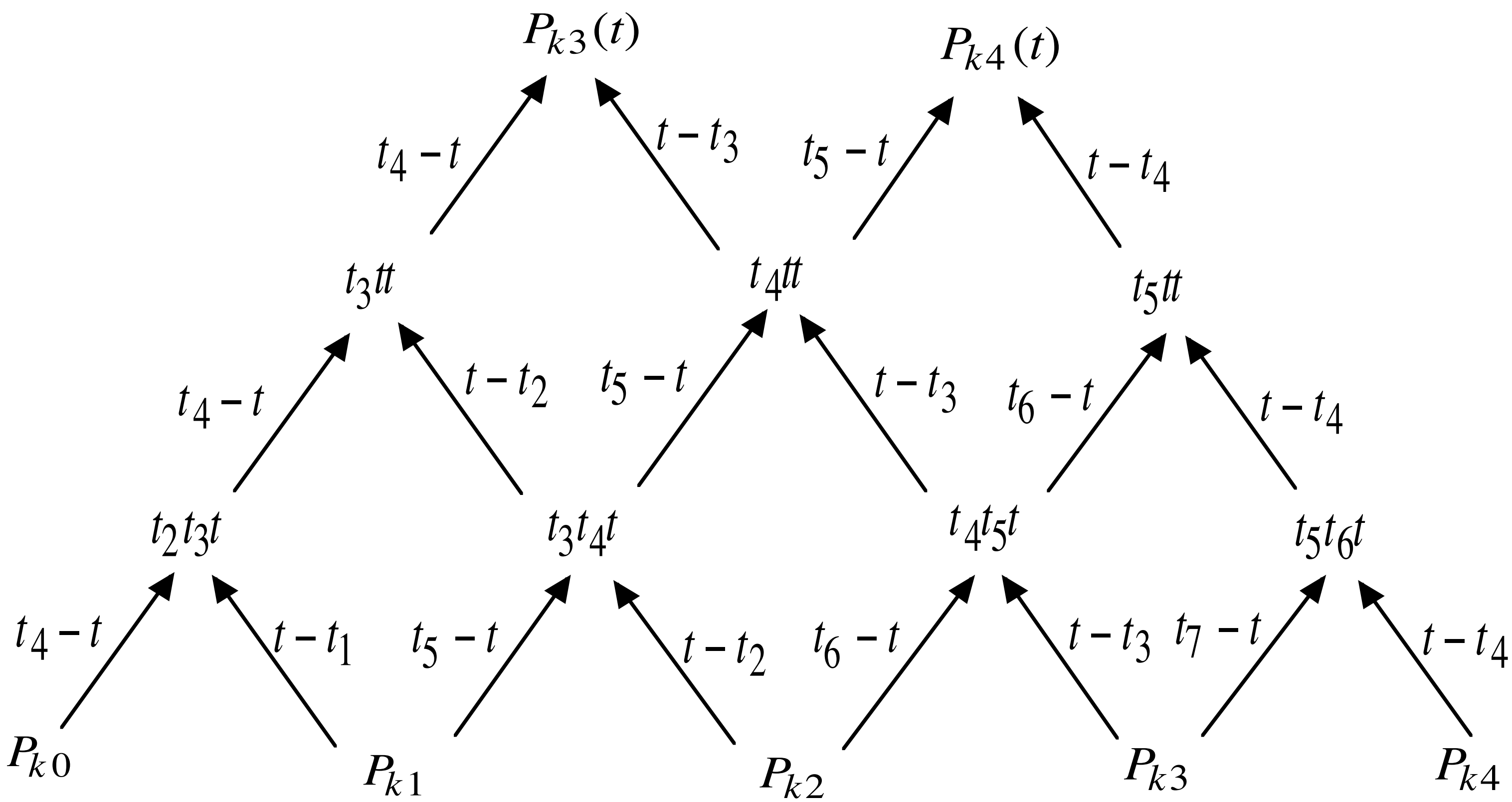
*Tensor Product B-spline Surface*

# de Boor Algorithm -- Tensor Product B-Spline Surface




*B-spline Curves*

# B-spline Curve



# Tensor Product B-Spline Surfaces

*Properties*

1. Piecewise polynomial
2. Continuity of order $C^{n-\mu}$ at knot lines of multiplicity $\mu$ on surfaces of degree $n$
3. Local control
4. Translation Invariance
5. Local convex hull property
6. Interpolation at knot lines where the multiplicity $\mu$ is equal to the degree $n$

*Knot Line Insertion Algorithms*

- Insert the same knots into each B-spline curve $P_k(t)$