Design und Evaluation einer integrierten Softwareplattform zur Optimierung von Stahlbeton-Tragwerken unter Einsatz evolutionärer Algorithmen

Dem Fachbereich Bauwissenschaften
der Universität Duisburg-Essen
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)
vorgelegte und genehmigte

DISSERTATION

von
Dipl.-Ing. Torben Jörg Pullmann
aus Groß-Zimmern

Essen 2011

Dipl.-Ing. Torben Jörg Pullmann

Der Lebenslauf ist in der Online-Version aus Gründen des Datenschutzes nicht enthalten.

Referent:	Univ.-Prof. Dr.-Ing. Martina Schnellenbach-Held
Korreferent:	Univ.-Prof. Dr.-Ing. Georg Thierauf
Tag der Einreichung:	07.01.2011
Tag der mündlichen Prüfung:	06.05.2011

Vorwort

Die vorliegende Arbeit entstand im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Massivbau der Technischen Universität Darmstadt sowie der Universität Duisburg-Essen.

Frau Univ.-Prof. Dr.-Ing. Martina Schnellenbach-Held danke ich herzlich für die Förderung und Betreuung dieser Arbeit sowie die umfangreiche Unterstützung in jeder Hinsicht während meiner gesamten Tätigkeit.

Herrn Univ.-Prof. Dr.-Ing. Georg Thierauf danke ich sehr für die Übernahme des Korreferats sowie sein großes Interesse an dieser Arbeit.

Professor Tomasz Arciszewski danke ich sehr für die angenehme Zusammenarbeit, die Förderung meiner Forschungstätigkeit und die Gastfreundschaft während meines Forschungsaufenthaltes an der George Mason University, Fairfax.

Großer Dank gilt meinen Kolleginnen und Kollegen für die angenehme und freundschaftliche Zusammenarbeit. Einen wesentlichen Beitrag zum Gelingen dieser Arbeit trugen auch die vielen fachlichen Diskussionen bei, für die ich mich insbesondere bei Peer Lubasch, Markus Hartmann und Mark Freischlad herzlich bedanken möchte.

Besonders danken möchte ich meiner Verlobten Beate sowie meiner Familie für den Rückhalt und die Unterstützung in den letzten Jahren.

Essen, im Januar 2011 Torben Pullmann

"Wer hohe Türme bauen will,
muss lange beim Fundament verweilen."

Anton Bruckner, österreichischer Komponist, 1824-1896

Inhaltsverzeichnis

Verzeichnis der Begriffe und Abkürzungen

Entwurf

Bauteil	Strukturbauteil eines Bauwerksmodells, beispielsweise eine Stütze oder ein lokal begrenzter Teil einer Deckenplatte.
Detailliertes Design	Feingranulare Definition detaillierter Bauwerksteile z.B. einzelner Strukturelemente (Wände, Stützen, Decken) sowie deren (Vor-)dimensionierung.
Element	Logische Einheit innerhalb eines Produktmodells. Im Rahmen dieser Arbeit beispielsweise Strukturbauteile (Stütze, Wand, etc.) oder logische Gruppierungen wie Stockwerke oder Stützensystemgruppen
FE-Element	Nach der Methode der finiten Elemente ein mathematisch definierter Teil eines Strukturbauteils
HLS	Fachplanung Heizung, Lüftung Sänitär
Konzeptuelles Design	Entwurfsphase zur Entwicklung und Definition konzeptueller Eigenschaften eines Produkts oder Bauwerks, beispielsweise Dimensionierung der Gebäudehülle oder grobe Aufteilung von Stockwerken und Räumen.
Produktmodell	Für den betrachteten Kontext adäquates Modell eines Produkts, beispielsweise eines Bauwerks.
Topologie, Topologiemodell	Beschreibung eines qualitativen, primitiven Geometriemodells. Im Rahmen der vorliegenden Arbeit beschreibt das Topologiemodell die Bauwerksgeometrie auf Ebene eines dreidimensionalen Achsmodells.
TWP	Tragwerksplanung

Evolutionäre Algorithmen

Bewertungsfunktion	Bewertung der Güte eines Individuums hinsichtlich festgelegter Bewertungskriterien
Bias	Differenz zwischen normalisierter Fitness und Reproduktionswahrscheinlichkeit
Bildungsvorschrift	Definition möglicher Ausprägungen und Strukturen der genotypischen Repräsentation von Individuen
BNF	Backus-Naur Form
BNF-Regel	Einzelne Grammatikregel innerhalb der BNF

Diversität	Maß für die Vielfalt der Ausprägung genetischer Merkmale innerhalb einer Population
EA	Evolutionäre Algorithmen
ES	Evolutionsstrategien
Exploitation	Verwertung, Nutzbarmachung - in Zusammenhang mit EA die Verfeinerung der durch Exploration gefundenen Lösungen
Exploration	Erkundung, Erforschung - in Zusammenhang mit EA die breit angelegte Suche innerhalb eines Problemraumes
Fitness	Funktional durch die Zuweisungsfunktion angepasstes Maß der Güte eines Individuums
Fitnessfunktion	Gesamtheit aus Bewertungsfunktion und Zuweisungsfunktion
GA	Genetische Algorithmen
Generation	Gruppe von Individuen, die sich im zeitlichen Optimierungsverlauf auf derselben Stufe befinden
Genotyp	Zusammenschluss der Erbinformationen eines Individuums
Genpool	Gesamtheit aller genetischen Variationen einer Population
GP	Genetische Programmierung
Individuum	Kleinste bewertbare Einheit des Optimierungsprozesses
Initialpopulation	Population mit Individuen der ersten Generation zu Beginn eines Optimierungsprozesses
nicht-terminiertes Symbol	Knoten der BNF, der eine beliebige Anzahl untergeordneter Knoten enthält
Optimierungsparameter	Parameter zur Steuerung des Optimierungsverlaufes
Optimierungsvariable	Informationseinheit innerhalb des Genotypen, die durch zufällige Initialisierung sowie durch genetische Operationen veränderlich ist
Phänotyp	Ausprägung der Merkmale eines Individuums, basierend auf seinem Genotyp
Population	Zusammengehörige, sich über mehrere Generationen verändernde Gruppe von Individuen
Problemraum	Mehrdimensionale Repräsentation der Gesamtheit aller möglichen Lösungen eines gegebenen Optimierungsproblems

Selektion (EA)	Prozess der Auswahl einzelner Individuen einer Population zur weiteren Verarbeitung, beispielsweise durch Rekombination, Mutation oder einfache Übertragung in eine Folgegeneration
Selektionsdruck	Stärke der Auswirkung des Fitnesswertes auf die Selektion einzelner Individuen
Spread	Differenz zwischen den Anzahlen erwarteter und tatsächlich selektierter Individuen
terminiertes Symbol	Endknoten der BNF mit fest definiertem Inhalt
Wertebereich-Symbol	Endknoten der BNF mit variablem, numerischem Inhalt innerhalb eines festgelegten Definitionsbereiches
Zuweisungsfunktion	Funktion zur Überführung der Bewertung eines Individuums in dessen Fitness.

Wissensbasierte Systeme

Berechnungselement	Element einer Wissensbank, das zur Berechnung von Parametern zur Anwendung kommt
Fachdisziplin	Spezialisierter Teilbereich einer Wissensdomäne
Inferenz, Inferenzprozess	Prozess der Schlussfolgerung und Lösungsfindung anhand einer Wissensbasis, bezogen auf ein konkretes Modell, beispielsweise ein Produktmodell
Inferenzmaschine	Softwarekomponente zur technischen Abbildung und Ausführung eines Inferenzprozesses
lose Kopplung	Technisch unabhängige Definition einzelner (Software-) komponenten oder untergeordneter Einheiten, die erst bei gemeinsamer Anwendung anhand von Metainformationen Abhängigkeiten ausbilden.
Selektion (Wissensverarbeitung)	Berechnungselement, das als Startpunkt eines Inferenzprozesses dient, z.B. zum Zwecke der Bemessung und Nachweisführung (Nachweis-Selektion) oder zum Entwurf (Design-Selektion)
Wissensbank	Softwaretechnische Abbildung einer Wissensbasis, beispielsweise in Form einer Datenbank
Wissensbasis	In sich schlüssige, evtl. lose gekoppelte Sammlung von Regeln und Berechnungselementen,
Wissensdomäne, Domäne	Abgegrenzter Wissensbereich, beispielsweise „Hochbau“ oder spezifischer „Tragwerksplanung“

1 Einleitung

1.1 Motivation

Die ingenieurmäßige Entwicklung komplexer Güter, seien es Maschinen, Fahrzeuge oder auch Bauwerke, ist ein schwieriger, iterativer Prozess. Bei der Gestaltung seriengefertigter Industrieprodukte wird der Entwurfsprozess meist durch Prototypen zur Optimierung der Produkte sowie der Produktionsabläufe unterstützt und verifiziert. Hierdurch können Fehlentscheidungen, die während der Entwurfsphase getroffen wurden, in späteren Iterationsschritten korrigiert werden. Dies gilt in der Regel nicht für den Entwurf von Bauwerken. Daher ist eine sorgfältige und nachhaltige Planung entscheidend für den Erfolg eines Bauprojektes, von der wirtschaftlichen Herstellung über die gesamte Nutzungsdauer bis hin zum Rückbau.

Stahlbeton-Bauwerke im Speziellen sind meist Unikate, die unter Beachtung einer immensen Fülle von Randbedingungen gestaltet werden müssen. Diese Randbedingungen beinhalten bauliche Gegebenheiten, behördliche und gesetzliche Vorgaben, Anforderungen, Möglichkeiten und Wünsche des Auftraggebers sowie das Bestreben nach Optimierung der qualitativen, ästhetischen und insbesondere der wirtschaftlichen Gestaltung des Bauwerkes. Unter Beachtung der prognostizierten Lebens- und Nutzungsdauer eines Stahlbeton-Bauwerks sind hierbei auch Gesichtspunkte der Nachhaltigkeit, mögliche Nutzungsänderungen, Umweltverträglichkeit, Optimierung des Energiebedarfs und wirtschaftliche Rückbau- und Verwertungsmöglichkeiten in die Planung mit einzubeziehen. Die Vereinbarung all dieser Aspekte stellt für die beteiligten Fachplaner eine sehr komplexe Aufgabe dar.

Moderne Computersysteme zur Unterstützung detaillierter Planungsphasen sind inzwischen unverzichtbar geworden, jedoch erfolgen die Entwürfe früher Planungsphasen meist manuell, teilweise mittels Computerunterstützung in der Zeichnungsdarstellung, jedoch ohne weit reichende Entscheidungsunterstützung. Gerade in frühen Planungsphasen, wenn die Auswirkungen von Planungsentscheidungen oft noch nicht direkt offensichtlich sind, haben Fehlentscheidungen meist die größten Auswirkungen auf den weiteren Erfolg des Gesamtprojektes. Daher ist eine frühest mögliche Computerunterstützung sinnvoll und notwendig.

Bedingt durch die Komplexität eines iterativen, mehrstufigen Planungsprozesses [CLA05] ist dieser geprägt durch die Erstellung einer Vielzahl von Planungsvarianten, die nicht alle zielführend sind. Die Evaluierung derartiger Varianten durch die beteiligten Fachplaner ist jedoch zeitaufwendig und daher kostenintensiv, wodurch die Anzahl der Varianten unter ökonomischen Gesichtspunkten beschränkt sein muss. Ein sehr gutes Werkzeug zur Unterstützung des Planungsprozesses wäre somit ein Tool, das die Gesamtmenge aller möglichen Planungsvarianten untersucht und daraus die besten Lösungen auswählt. Alleine durch die Anzahl der veränderlichen Parameter ergibt sich jedoch bereits für mittlere Bauprojekte ein unendlich großer Suchraum, der auch von den aktuell schnellsten Computersystemen nicht in adäquater Rechenzeit beherrschbar wäre.

Für den beschriebenen Fall bieten sich automatisierte Optimierungsverfahren an, allen voran evolutionäre Algorithmen. Die Unvoreingenommenheit eines automatisierten Optimierungsverfahrens ermöglicht die wirtschaftliche Evaluierung auch zunächst ungewöhnlich erscheinender Lösungen, die ein Ingenieur möglicherweise nicht weiter verfolgen würde, die jedoch in ihrer Gesamtheit zu guten Ergebnissen führen können.

Die Optimierung von Tragstrukturen erfordert die Betrachtung sowohl kontinuierlicher als auch diskreter Parameter [THI96]. Diskrete Parameter repräsentieren topologische sowie konzeptuelle Designinformationen. Diese können sehr unterschiedliche Ausprägungen annehmen, beispielsweise die Wahl geeigneter Deckensysteme oder die Position und Anzahl zusammengesetzter aussteifender Bauteile. Kontinuierliche Parameter definieren detaillierte Merkmale des Tragwerks, beispielsweise geometrische Abmessungen. Unter „kontinuierlich“ wird in diesem Zusammenhang auch eine hinreichend genaue Form der Diskretisierung einzelner Parameter verstanden.

Die Kombination konzeptueller und detaillierter Designparameter in einer gemeinsamen zu optimierenden Beschreibungsdefinition ist notwendig, da die zur Optimierung notwendige Bewertung eines Tragwerks nicht anhand unvollständiger Informationen erfolgen kann. Gleichzeitig ergeben sich hieraus einige Schwierigkeiten in Bezug auf die zu wählende Repräsentationsform. Die Variabilität des konzeptuellen Designs führt zu einer differierenden Anzahl notwendiger Parameter, wodurch keine fixe Länge der Codierung angenommen werden kann. Genetische Operationen, die erfolgreich auf konzeptuelle Informationen anwendbar sind, können für kontinuierliche Parameter ein grundsätzlich abweichendes Verhalten hervorrufen. Der inverse Sachverhalt gilt gleichfalls. Im Rahmen der vorliegenden Arbeit sollen praktikable Konzepte zur geeigneten Repräsentation konzeptueller und detaillierter Designparameter mit dem Ziel eines praktikablen und performanten Optimierungsprozesses entwickelt werden.

Die praktische Anwendbarkeit eines Verfahrens zur automatisierten Optimierung von Tragstrukturen ist abhängig davon, ob die signifikanten Ziele des Planers tatsächlich vollständig erfasst werden. Einige der bekannten Ansätze zur Optimierung von Bauwerksentwürfen betrachten ausschließlich die Minimierung des Eigengewichtes einer gegebenen Tragstruktur. Das Gewicht wird als vereinfachter Indikator für die gesamten Herstellkosten eines Bauwerks angesehen. Unbeachtet bleiben hierbei weitere kritische Aspekte, wie nutzbare Grundfläche, räumliche Aufteilung und Zweckmäßigkeit der Grundrisse oder die Folgekosten von Ausbaugewerken. Eine reine Gewichtsoptimierung der Tragstruktur wird daher als vollkommen ungenügend angesehen.

Die Fülle möglicher Randbedingungen lässt sich in einer standardisierten Softwarelösung kaum repräsentieren, was als einer der Hauptgründe für den geringen Einsatz von Optimierungsverfahren im Hochbau angesehen werden kann. Die vorliegende Arbeit verfolgt daher weiterhin das Ziel, ein durch den Anwender sehr gut auf die gegebenen Randbedingungen adaptierbares Softwarekonzept zu entwickeln.

1.2 Stand der Forschung

Eine Reihe unterschiedlicher Forschungsprojekte beschäftigt sich mit der Unterstützung des Entwurfs von Tragwerken. Ein weit verbreiteter Ansatz ist die Optimierung kontinuierlicher Designparameter [RAM00], [BLE02]. Voraussetzung dieses Ansatzes ist der manuelle konzeptuelle Entwurf einer geeigneten Tragstruktur, die wesentlich die Effizienz des gesamten Tragwerks bestimmt. Der Optimierungsprozess hat hier auf die Wahl des Tragsystems keinerlei Einfluss, wodurch erhebliche Einsparpotentiale vernachlässigt werden.

Arciszewski und De Jong stellen in [ARC01] ein Konzept zur konzeptuellen und detaillierten Optimierung von Stahl-Rahmentragwerken vor. Das System variiert Form, Anzahl und Position von aussteifenden Elementen, die Art der Rahmenanschlüsse (biegesteif / gelenkig) sowie einzelne Querschnittswerte. Zur Bewertung der Trag-

werke wird ausschließlich das Eigengewicht der Tragstruktur herangezogen. Das System idealisiert die Tragwerke durch manuelle Definition der vertikalen und horizontalen Tragglieder sowie durch Beschränkung des Modells auf eine vertikale Schnittebene.

In [GEY07] wird ein multidisziplinärer Ansatz zur mehrkriteriellen Tragwerksoptimierung vorgestellt. Am Beispiel einer Halle werden ökonomische, ökologische, funktionale und ästhetische Kriterien zur Optimierung der Tragstruktur sowie maßgebender Designparameter verwendet. Die Bewertung automatisiert berechenbarer Kriterien erfolgt getrennt von der manuellen Bewertung der Ästhetik in einem eigenen Optimierungsschritt.

Ein GA-basiertes System für den konzeptionellen Gebäudeentwurf wird von Rafiq et al. in [RAF99] vorgestellt. Es handelt sich hierbei um ein hybrides System, das zur Bewertung der Entwürfe ein entsprechend trainiertes neuronales Netz verwendet. Dieses System berücksichtigt nicht das Aussteifungssystem, was für den Entwurf von Hochhäusern notwendig ist.

Das von Grierson et al. in [GRI02] vorgeschlagene System ermöglicht die Optimierung von Bürogebäuden unter Berücksichtigung der Tragstruktur, Geschosshöhen und –anzahl und der Grundfläche. Die multikriterielle Optimierung bewertet neben den Herstellungskosten die Betriebskosten sowie die erzielbaren Einnahmen. Dieses System ist allerdings auf vorgegebene Grundrisse in sämtlichen Geschossen beschränkt.

In [SAR00] wird von Sarma und Adeli ein mehrstufiges Optimierungskonzept unter Einsatz des Fuzzy Augmented Lagrangian GA vorgestellt. Hierbei wird zunächst eine Stahl-Fachwerkkonstruktion hinsichtlich des Gewichts optimiert. In einem zweiten Schritt erfolgt mittels eines regelbasierten Fuzzy-Systems die Selektion der Art und Anzahl geeigneter Lieferprofile unter Beachtung der Kriterien minimales Gewicht, minimale Querschnittsanzahl und minimale Gesamtkosten.

Yang und Soh stellen in [YAN02] ein System zur konzeptuellen und detaillierten Optimierung von Tragstrukturen basierend auf genetischer Programmierung vor. Ihr System zeigt klar die Vorteile der genetischen Programmierung im Vergleich zu traditionellen Optimierungsverfahren. Die variable Codelänge bietet die Möglichkeit, unterschiedliche konzeptuelle Entwürfe zu repräsentieren und gleichzeitig in den einzelnen Entwürfen detaillierte Entwurfsparameter zu berücksichtigen.

Die vorgenannten Arbeiten verdeutlichen die grundsätzliche Eignung evolutionärer Algorithmen zur Optimierung komplexer Tragstrukturen, berücksichtigen jedoch nicht alle zur Optimierung von Hochhäusern erforderlichen Randbedingungen. Keines der genannten Systeme ist in der Lage, effizient innovative dreidimensionale Tragstrukturen zu erzeugen. Für den praxisrelevanten Einsatz ist ein hoher Grad der Adaptierbarkeit erforderlich, der ebenfalls in keinem der bisher entwickelten Ansätze berücksichtigt wurde.

1.3 Zielsetzung dieser Arbeit

Die vorliegende Arbeit verfolgt das Ziel, ein skalierbares und flexibles Softwaresystem zu entwerfen, das durch Anwendung unterschiedlicher Methoden der künstlichen Intelligenz ermöglicht, unter Einhaltung von gegebenen Randbedingungen den konzeptuellen und detaillierten Entwurf von komplexen Stahlbeton-Tragwerken zu optimieren. Besonderer Wert wird auf die universelle und interdisziplinäre Anpassbarkeit des Grundsystems, der Zielfunktionen sowie der Randbedingungen und der zugrun-

de liegenden Regelwerke gelegt. Der Anwender soll in die Lage versetzt werden, durch Festlegung von bekannten Vorgaben und Regeln vorab den Suchraum einzuschränken und möglichst präzise unmögliche und ungewollte Varianten auszuschließen. Das System soll nicht auf die Disziplin des Stahlbetonbaus beschränkt sein, sondern möglichst umfassend die relevanten Belange und Optimierungspotentiale verschiedener Fachdisziplinen berücksichtigen können. Beispielhaft seien hier genannt:

- Architektur
- Tragwerksplanung
- Fachplanung Heizung, Lüftung, Sanitär
- Fachplanung Elektrik
- Fachplanung Brandschutz
- Fachplanung Schallschutz
- Fachplanung Wärmeschutz
- Facility Management

Die Evaluierung der entwickelten Konzepte und prototypischen Software-Tools erfolgt anhand von 3D-Modellen von Stahlbeton-Hochhäusern. Diese Bauwerke sind besonders interessant, da durch geschickte Wahl und Dimensionierung von Geometrien und Konstruktionselementen hohe Einsparpotentiale erzielt werden. Gleichzeitig müssen jedoch die Nutzbarkeit und die Standsicherheit in statischer- und dynamischer Hinsicht (Wind- und Erdbebenbelastung) gewährleistet sein.

2 Grundlagen

2.1 Evolutionäre Algorithmen

"Alles, was gegen die Natur ist,
hat auf Dauer keinen Bestand."

Charles Darwin, 1809 - 1882

2.1.1 Biologische Evolution

Evolutionäre Algorithmen basieren auf dem Vorbild der biologischen Evolutionstheorie [DAR1884]. Diese beschreibt die allmähliche Veränderung der genetischen Information bei Lebewesen mit dem Ziel der Anpassung an die gegebenen und sich kontinuierlich ändernden Bedingungen ihrer Umwelt. In der Natur sind dies primär der Lebensraum, dessen klimatische Bedingungen und die zur Verfügung stehenden Nahrungsressourcen. Ebenso sind die Verbreitung der eigenen Spezies sowie das Verhältnis zu Fressfeinden entscheidend für den Evolutionsprozess. Beispielsweise können Fluchttiere enorme Geschwindigkeiten erreichen, andere Arten sind durch perfekte Tarnung in ihrer natürlichen Umgebung für Feinde nahezu unsichtbar. Die meisten evolutionären Anpassungen lassen sich auf das Ziel der Sicherstellung des Überlebens einer Spezies zurückführen.

Ein wesentlicher Aspekt der biologischen Evolution ist die Repräsentation der Individuen anhand ihres genetischen Codes, der als ein biologischer „Bauplan“ für die Entwicklung von Zellen angesehen werden kann. Die genetische Information liegt in Form der „Desoxyribonukleinsäure“ (kurz „DNS“, oft auch „DNA“ nach dem englischsprachigen Äquivalent „deoxyribonucleic acid“) vor [WAT68]. Die DNS besteht im Wesentlichen aus einem Polymer, das sich aus Desoxyribonukleotiden zusammensetzt. Diese lassen sich in vier Arten unterteilen: Adenin (A), Cytosin (C), Guanin (G) und Thymin (T), die jeweils mit einem Phosphorsäure-Molekül verbunden sind. Durch Kombination dieser Nukleotiden ergeben sich beispielsweise für den genetischen Code des Menschen mehrere Billionen Kombinationsmöglichkeiten. Der genetische Code wird als „Genotyp“, die tatsächliche Ausprägung der daraus resultierenden Individuen als „Phänotyp“ bezeichnet.

Der biologische Evolutionszyklus lässt im Wesentlichen drei Mechanismen erkennen, die zur Anpassung der genetischen Information führen: Selektion, Rekombination und Mutation.

- Selektion: Die Selektion ist das treibende Steuerungsinstrument des Evolutionsprozesses. Sie bewirkt, dass Individuen mit einer geringen „Fitness“, was mit einer schlechten Anpassung an ihre Umwelt assoziiert wird, mit einer hohen Wahrscheinlichkeit von höherwertigen Individuen dominiert und verdrängt werden. Hierdurch wird erreicht, dass sich nur die „fittesten“ Individuen fortpflanzen und deren Erbgut somit die Grundlage der folgenden Generation bildet („Survival of the fittest“, [DAR1894]). Die Selektion ist durch Bestimmung der Fitness eines Individuums das einzige evolutionäre Prinzip, dem zielgerichtetes Vorgehen zugrunde liegt.

- Rekombination: Bei der Rekombination wird durch sexuelle Fortpflanzung von Eltern-Individuen deren genetische Information vermischt, wodurch neue Individuen entstehen. Das Erbgut der Nachkommen besteht jeweils zu einem bestimmten Teil aus dem Erbgut der Eltern-Individuen, stellt jedoch nach der Rekombination wieder eine vollständige Einheit genetischer Informationen dar.
- Mutation: Dies ist eine zufällige, meist geringfügige Veränderung der genetischen Information einzelner Individuen. Während die Rekombination ausschließlich vorhandene Informationen innerhalb des Genpools verwendet, wodurch die Diversität eingeschränkt wäre, führt die Mutation zu vollständig neuen Informationen, die nicht durch Vererbung herbeigeführt wurden.

Durch die Vereinigung der beschriebenen Mechanismen wird erreicht, dass im Laufe des biologischen Evolutionsprozesses die Individuen stetig besser an ihre Umwelt und weitere Randbedingungen angepasst werden und sich dem theoretisch, wenn auch nicht erreichbaren „optimalen Individuum“ annähern. Auch allmähliche oder plötzliche Veränderungen der Umweltbedingungen während des Entwicklungsprozesses werden somit berücksichtigt und führen im Laufe der Zeit zu entsprechend angepassten Individuen.

2.1.2 Anwendung auf technische Optimierungsprobleme

Die Prinzipien der biologischen Evolution sind auf eine ganze Reihe von Optimierungsproblemen aus nahezu allen Wissensdomänen übertragbar, die durch traditionelle Optimierungsverfahren aufgrund von Nichtlinearitäten, Diskontinuitäten und Multimodalität nicht hinreichend lösbar sind [HEM02]. Voraussetzung zur Anwendung evolutionärer Optimierung ist zunächst die Möglichkeit einer geeigneten Repräsentationsform der Individuen, dem Genotypen. Ebenso ist eine Methode zur Umwandlung des Genotypen in einen bewertbaren Phänotypen notwendig. Die Form der gewählten genotypischen Repräsentation hat hierbei einen enormen Einfluss auf das Verhalten des Optimierungsprozesses. Eine ungeeignete Repräsentationsform kann unter Umständen dazu führen, dass eine hohe Anzahl möglicher Individuen der gewählten Repräsentation als nicht lösbar angesehen werden muss, was den Suchraum unnötig erweitert und die Effizienz eines evolutionären Algorithmus enorm reduzieren kann.

Weiterhin erforderlich ist die Bereitstellung einer oder mehrerer geeigneter Methoden zur Bewertung der Individuen anhand der zu optimierenden Kriterien, der so genannten „Fitnessfunktion“. Die Fitness dient als Vergleichskriterium der Individuen untereinander in Bezug auf die gegebenen Randbedingungen des Optimierungsproblems. Das Ergebnis der Fitnessfunktion sollte einen kontinuierlichen Fitnesswert liefern, der die Individuen im Umfeld des gegebenen Problems vergleichbar macht und bei Veränderungen die Stärke der jeweiligen Änderung adäquat reflektiert. Je höher der Grad der Diskontinuität dieser Fitnessfunktion ist, desto eher entspricht die genetische Optimierung letztendlich einer rein zufälligen Suche, was beispielsweise bei dem Versuch der Entschlüsselung von Passworten[1] der Fall ist. Hier liefert die Fitnessfunktion lediglich „ja“ oder „nein“, wodurch keine Optimierung möglich ist.

[1] Passworte werden oft durch die Methode der „Brute force“ (engl. für „brutale Gewalt“) ermittelt. Durch einfaches Ausprobieren sämtlicher möglicher Passworte kann somit ein Passwort ermittelt werden, wenn auch diese Methode abhängig von der Passwort-Länge sehr aufwändig ist.

Die resultierende Größe der Fitnessfunktion, die „Fitness“ eines Individuums, wird bei evolutionären Algorithmen primär zur Steuerung der Selektion herangezogen. Die Mechanismen der Rekombination und Mutation bedienen sich der durch Selektion erworbenen Individuen einer Generation, werden aber im Allgemeinen nicht weiter durch deren Fitness beeinflusst. Alleine die Selektion stellt somit mittels Anwendung des Fitnesswertes die Schnittstelle zwischen der Optimierungsstrategie und dem gegebenen Problemraum dar.

Der Begriff der „Evolutionären Algorithmen“, im englischen Sprachraum bekannt als „Evolutionary computing“, vereint die Bereiche „Evolutionsstrategien“, „Evolutionäre Programmierung“, „Genetische Algorithmen“ und „Genetische Programmierung“, die in den Kapiteln 2.1.3 bis 2.1.6 detaillierter erläutert werden.

2.1.2.1 Fitness-Zuweisung

Evolutionäre Algorithmen verwenden zur Bewertung der Individuen eine Fitnessfunktion. Diese setzt sich im Allgemeinen zusammen aus einer problemspezifischen, mathematischen Bewertungsfunktion sowie einer Zuweisungsfunktion. Das Ergebnis der Bewertungsfunktion, der Zielfunktionswert $Z_{(i)}$ eines Individuums *i* wird durch die Zuweisungsfunktion in einen Fitnesswert $F_{(i)}$ überführt. Das primäre Ziel der Zuweisungsfunktion ist es, den Selektionsdruck *P* in einem günstigen Bereich zu stabilisieren. Dieser gibt an, wie stark der Zielfunktionswert die Fortpflanzungswahrscheinlichkeit der Individuen beeinflusst. Ein zu hoher Selektionsdruck führt zu vorzeitiger Konvergenz, da die besten Individuen sich überproportional schnell vermehren und innerhalb weniger Generationen die gesamte Population einnehmen. Ein zu geringer Selektionsdruck hingegen resultiert in einer nahezu zufälligen Suche innerhalb des Selektionspools, so dass keine Optimierung stattfinden kann.

Grundsätzlich wird zwischen proportionaler Zuweisung und reihenfolgebasierter Zuweisung (Ranking) unterschieden. Bei proportionaler Zuweisung erfolgt die Berechnung des Fitnesswertes ausschließlich basierend auf dem Zielfunktionswert Z_i des einzelnen Individuums ((2-1) - (2-4)). Parameter *a*, *b* und *k* sind festzulegende Konstanten.

In (2-2) erfolgt die dynamische Anpassung des Fitnesswertes durch Variation des Parameters b_{gen} als Funktion der Generation. Auch wenn der zu erwartende Bereich der Zielfunktionswerte bekannt ist, besteht bei proportionalen Zuweisungsfunktionen keine effektive Kontrolle des resultierenden Selektionsdrucks. Vorteilhafter sind daher reihenfolgebasierte Zuweisungsfunktionen (2-7). Hierbei bezeichnet Pos_i die Position des jeweiligen Individuums innerhalb des nach Z_i sortierten Selektionspools mit n_{ind} Individuen. Das schlechteste Individuum erhält die Position $Pos_i=1$, das Beste $Pos_i=n_{ind}$. Im Fall des linearen Ranking ist zu beachten, dass der vorgegebene Selektionsdruck *P* nicht größer als 2.0 gewählt werden darf. Die Unbekannte *X* in (2-5) wird durch Lösung des Polynoms in (2-6) bestimmt.

lineare Skalierung: $$F_i = a \cdot Z_i + b \quad \textbf{(2-1)}$$

dynamische Skalierung: $$F_i = a \cdot Z_i + b_{gen} \quad \textbf{(2-2)}$$

logarithmische Skalierung: $$F_i = b - \log(Z_i) \quad \textbf{(2-3)}$$

exponentielle Skalierung: $$F_i = (a \cdot Z_i + b)^k \quad \textbf{(2-4)}$$

nichtlineares Ranking:

$$F_i = \frac{n_{ind} \cdot X^{Pos_i - 1}}{\sum_{i=1}^{n_{ind}} X^{i-1}} \quad \textbf{(2-5)}$$

$$0 = (P-1) \cdot X^{n_{ind}-1} + P \cdot X^{n_{ind}-2} + \ldots + P \cdot X + P \quad \textbf{(2-6)}$$

lineares Ranking:

$$F_i = 2 - P + 2 \cdot (P-1) \cdot \frac{Pos_i - 1}{n_{ind} - 1} \quad \textbf{(2-7)}$$

2.1.2.2 Selektion

Im Bereich der evolutionären Optimierung bezeichnet die Selektion den Prozess der Auswahl geeigneter Individuen aus einem Selektionspool einer Generation, mit dem Ziel der Bildung einer weiteren Generation. Hierzu existieren eine ganze Reihe unterschiedlicher Verfahren [POH99].

Die einfachste, auf der Fitness eines Individuums basierende Selektion ist die *Rang-basierte Selektion*. Hierbei werden die Individuen entsprechend ihres Ranges sortiert, und die *n* besten Individuen selektiert. Dieses Verfahren wird im Allgemeinen nicht eingesetzt, da bereits nach wenigen Generationen nur noch Merkmale der ursprünglich besten Individuen in der Population vorhanden sind und die Diversität durch verstärkte Mutation erhalten werden muss.

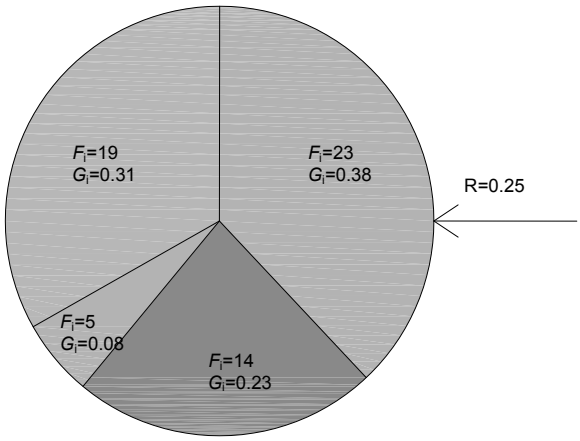


Abbildung 2-1: Rouletteselektion

Das gängigste Selektionsverfahren ist die *Rouletteselektion* (engl. *„roulette wheel selection“*). Die Bezeichnung erhielt dieses Verfahren aufgrund der Verteilung der Selektionswahrscheinlichkeiten entsprechend ihrer jeweiligen Fitness, analog zu einem Rouletterad (siehe Abbildung 2-1). Nach der Erzeugung einer Zufallszahl zwischen 0 und 1 (wobei dieser Bereich der Abdeckung des gesamten Rouletterads entspricht), wird ein Individuum selektiert, in dessen Bereich der Wert dieser Zufallszahl fällt. Hierdurch ist gewährleistet, dass bessere Individuen mit einer höheren Wahrscheinlichkeit selektiert werden, die schlechteren Individuen zur Erhaltung der

Diversität trotzdem eine reelle Chance auf Selektion erhalten. Der funktionale Zusammenhang zwischen dem Fitnesswert F_i und dem zugewiesenen Anteil G_i auf dem Rouletterad ergibt sich aus (2-8).

Rouletteselektion:

$$G_i = \frac{F_i}{\sum_1^n F_i} \qquad \textbf{(2-8)}$$

Ein der Rouletteradselektion sehr ähnliches Verfahren ist das *stochastic universal sampling* [POH99]. Die Verteilung der Anteile im Verhältnis zu den Fitnesswerten erfolgt unverändert, jedoch wird an Stelle einer Zufallszahl eine Gruppe von Zeigern in der Anzahl *n* der zu selektierenden Individuen gleichmäßig auf dem Rouletterad verteilt. Die Anwendung der Zufallsvariablen *R* erfolgt durch synchrone Verschiebung der zusammenhängenden Zeiger um den Wert R/n_{ind}. Im Gegensatz zur Rouletteradselektion garantiert das stochastic universal sampling einen minimalen *Spread*[2], welcher dem Bereich der möglichen Werte für die Anzahl der Nachkommen eines Individuums entspricht.

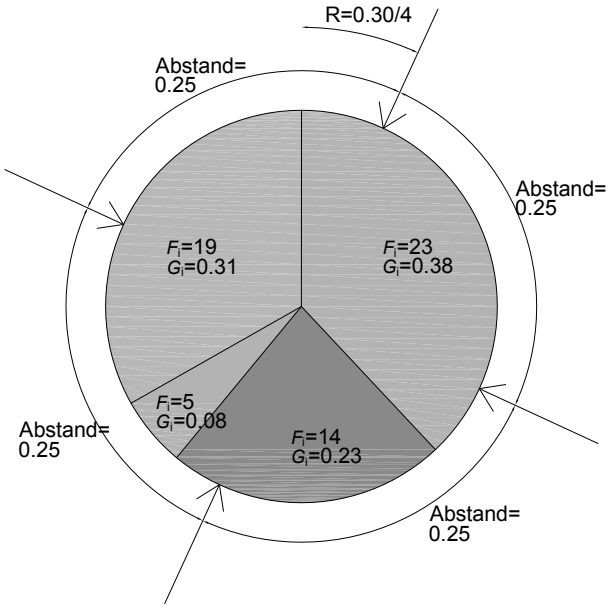


Abbildung 2-2: stochastic universal sampling

Bei der *Turnierselektion* (engl. *„tournament selection"*) werden zunächst einige Individuen zufällig aus einer Generation in einer Auswahlgruppe selektiert, woraufhin die Individuen dieser Gruppe in einer Art Turnier gegeneinander antreten. Der „Sieger" des Turniers ist immer das Individuum mit der besten Fitness. Vorteilhaft sind hierbei die leichte Implementierbarkeit sowie die relativ einfache Anwendung der Selektionsintensität durch direkte Steuerung der Größe der Auswahlgruppe. Je größer die

[2] *Spread* bezeichnet den Unterschied zwischen den Anzahlen erwarteter und tatsächlich selektierter Individuen

Auswahlgruppe gewählt wird, desto geringer sind die Chancen für schwächere Individuen. Enthält die Auswahlgruppe lediglich ein einziges Individuum, so bleibt die Fitness gänzlich unberücksichtigt. Dies ist auch der einzige Fall, in dem das insgesamt schwächste Individuum selektiert werden könnte.

Die *Boltzmann-Selektion* verwendet Prinzipien des Simulated Annealing (engl. für „Simulierte Abkühlung“) [BÄC97]. Grundlage dieser Selektion ist das Boltzman-Trial (2-9), eine Funktion, die den Wettbewerb zweier Individuen *i* und *j* anhand deren Fitness und einer im Verlauf variablen Temperatur *T* entscheidet. Die Temperatur sinkt mit der Zeit analog zum Abkühlvorgang eines glühenden Werkstückes, wodurch mit fortschreitenden Generationen die Fitness eines Individuums größeren Einfluss auf die Selektion nimmt.

Boltzman-Trial:

$$G_i = \frac{1}{1+e^{(F_i-F_j)/T}} \tag{2-9}$$

2.1.2.3 Rekombination

Die Rekombination simuliert bei Evolutionären Algorithmen das biologische Vorbild der geschlechtlichen Fortpflanzung. Bei der Rekombination werden die Merkmale zweier Individuen derart vereinigt, dass zwei neue, vollständige Individuen entstehen. Die Individuen einer Optimierungsmethode sind in der Regel geschlechtslos, weshalb die beiden Eltern einer Rekombination frei innerhalb der Population selektiert werden können.

Das Rekombinationsverhalten ist sehr stark abhängig von der gewählten Repräsentationsform. So können beispielsweise bei einer binären Repräsentation die genetischen Informationen an beliebiger Stelle aufgetrennt und ab dieser Stelle zwischen den Eltern-Individuen getauscht werden. Hierdurch kann auch innerhalb zusammenhängender Informationseinheiten an ungünstiger Stelle getrennt werden, wodurch „gute“ Informationen aus den Eltern verloren gehen können. Bei reell codierter Repräsentation verwendet die Rekombination oftmals Mittelwerte zwischen Informationen der Eltern. Andere Repräsentationsformen erlauben hier ein eher zielgerichtetes Vorgehen, beispielsweise die Genetische Programmierung.

Auch wenn die Rekombination durch Anwendung unterschiedlicher Eltern und Kreuzungspunkte eine hohe Anzahl an Kombinationen ermöglicht, so würde mit Fortschreiten des Optimierungsprozesses, bei alleiniger Anwendung von Selektion und Rekombination, die Diversität innerhalb des Genpools relativ schnell verloren gehen [BÄC97].

2.1.2.4 Mutation

Mutationen sind Veränderungen an der genetischen Information eines einzelnen Individuums. Meist sind diese Veränderungen von geringem Umfang, können jedoch abhängig von der Repräsentationsform auch größere Veränderungen hervorrufen. Im Falle der binären Repräsentation erfolgt die Mutation oftmals durch Negierung zufällig ausgewählter Bits des Individuums. Reell codierte Repräsentationsformen verwenden zur Mutation geringfügige Veränderungen einzelner Werte durch Addition oder Subtraktion eines geringen Wertes zum Ursprungswert.

2.1.2.5 Exploration vs. Exploitation

Evolutionäre Algorithmen basieren im Allgemeinen auf der Suche nach einer möglichst optimalen Lösung innerhalb eines n-dimensionalen Suchraums potentieller Lösungen. Evolutionäre Algorithmen haben sich im Vergleich zu klassischen, umfassenden Suchmethoden als effizienter erwiesen. Sie sind stochastische Methoden, die sich Phänomene der natürlichen Evolution zu Nutze machen. Der Suchraum enthält neben einer oder mehreren optimalen Lösungen jedoch häufig lokale Extrema. Stochastische Suchmethoden beinhalten die Gefahr, dass lokale Extrema überbewertet und gleichzeitig globale Extrema nicht erkannt werden. Zwei wesentliche Mechanismen bestimmen hierbei die stochastische Suche innerhalb des betrachteten Suchraums [HAL99]:

- Exploration ("Erkundung, Erforschung"): Breite Abdeckung des Suchraumes durch hohe Diversität innerhalb einer Population.
- Exploitation ("Verwertung, Nutzbarmachung"): Fokussierung auf die Individuen mit höherer Fitness durch Reduktion der Diversität.

Diese konträren Mechanismen sind für den erfolgreichen Einsatz evolutionärer Algorithmen gleichermaßen notwendig. Das Verhältnis zwischen Exploration und Exploitation, das über die Diversität steuerbar ist, erfordert besondere Beachtung. Die Diversität einer Population wird in EA indirekt durch die genetischen Operatoren, deren Anwendungshäufigkeit und Stärke bestimmt. Daher ist eine sorgfältige Wahl der Optimierungsparameter erforderlich. Eine Anpassung der Parameter während des Optimierungsfortschrittes kann sinnvoll sein, um zu Beginn einer Optimierung den Schwerpunkt auf Exploration zu legen, während nach hinreichender Erkundung des Suchraumes eine Anpassung der Parameter zur Verstärkung der Exploitation und damit zur Verfeinerung der gefundenen Lösungen führen kann.

2.1.2.6 Abbruchkriterium

Da die Qualität der erreichbaren Lösung im Vorfeld meist nicht bekannt ist, durchläuft die Optimierung mehrere Iterationsschritte, bis eines der vordefinierten Abbruchkriterien erfüllt wird. Im Allgemeinen wird als Abbruchkriterium das Erreichen eines Schwellwertes für den Zielfunktionswert des besten Individuums gewählt. Da nicht sicher ist, ob dieser Wert jemals erreicht wird, sollte zusätzlich eine Beschränkung der Anzahl durchlaufener Generationen oder durchgeführter Fitnesswertberechnungen erfolgen.

2.1.3 Evolutionsstrategien

Evolutionsstrategien (ES) wurden 1965 von I. Rechenberg und H.-P. Schwefel entwickelt [REC73], [SCHW95]. Ursprünglich waren Optimierungen hydrodynamischer Systeme das Ziel der Entwicklung, doch später wurden Evolutionsstrategien zur generellen Funktionsoptimierung erweitert.

Die einfachste Variante der Evolutionsstrategien verwendet zur Repräsentation der Individuen n-dimensionale, reell codierte Vektoren. Die Aufgabe der Optimierung besteht im Finden desjenigen Vektors x, der das globale Extremum einer gegebenen n-dimensionalen Funktion F(x) liefert.

Der Optimierungsprozess beginnt mit Bildung einer möglichst gleichmäßig über den Suchraum verteilten Initialpopulation von Individuen in Form von Vektoren. Zur Bildung einer neuen Generation werden aus einer Population mit μ Eltern λ Nachkommen erzeugt. Nachkommen entstehen ausschließlich durch Mutation der Individuen. Aus der Gesamtheit aus Eltern und Nachkommen ($\mu + \lambda$) werden mittels fitnessbasierter Selektion μ Individuen in die nächste Generation überführt. Daher wird diese Form der ES auch oftmals „($\mu + \lambda$) ES“ abgekürzt. Eine weitere Variante der ES ist „(μ, λ) ES“. Hierbei werden ebenso aus μ Eltern λ Nachkommen gewonnen. Aus der Menge λ der Nachkommen werden μ Individuen für die nächste Generation selektiert, woraus folgt, dass in diesem Fall $\lambda > \mu$ sein muss.

Es wird davon ausgegangen, dass genetische Transformationen immer geringe Veränderungen sämtlicher Informationen mit sich führen. Diese werden bei Evolutionsstrategien durch geringfügige Veränderungen sämtlicher Komponenten um einen Gauss-basierten Zufallswert simuliert. Der Erwartungswert *m* der Normalverteilung nach Gauss entspricht dem ursprünglichen zu mutierenden Wert. Die Standardabweichung σ gibt die Stärke der Mutation an.

Die Variation der Mutationsstärke σ hat einen großen Einfluss auf das Optimierungsverhalten. Ein zu groß gewählter Wert für σ ermöglicht eine breite Untersuchung des Suchraums, führt jedoch langfristig nicht zu der erforderlichen Feinabstimmung der Lösungen. Kleinere Werte für σ verzögern den Optimierungsvorgang zu Beginn und führen schnell zu lokalen Optima. Daher ist eine variable Anpassung der Mutationsstärke sinnvoll. Rechenberg formulierte zur Anpassung der Mutationsstärke die 1/5 Erfolgsregel: Wenn der Quotient der erfolgreichen Mutationen bezogen auf die Gesamtzahl der durchgeführten Mutationen je Generation größer als 1/5 ist, wird σ erhöht, ansonsten verringert [BÄC97] (vgl. auch Kap. 2.1.2.5).

Beeinflusst durch die Entwicklung genetischer Algorithmen durch Holland [HOL75] wurden Evolutionsstrategien um Operatoren zur Rekombination erweitert, wodurch die Unterscheidung zwischen Evolutionsstrategien und genetischen Algorithmen aufgeweicht wurde.

2.1.4 Evolutionäre Programmierung

L. Fogel nutzte eine Form simulierter Evolution zur Optimierung endlicher Automaten *(engl.: finite state machine, FSM)* [FOG66]. Ein endlicher Automat ist ein Modell der Automatentheorie zur Beschreibung von Aktionen, Zuständen und Zustandsübergängen. Fogel beschäftigte sich mit der Optimierung deterministischer Transduktoren. Diese Form endlicher Automaten ist in der Lage, basierend auf dem aktuellen Zustand des Automaten sowie eines Eingabestrings einen entsprechenden Ausgabestring zu erzeugen. Sowohl Eingaben als auch Ausgaben folgen einem strengen symbolischen Vokabular.

Tabelle 2-1: Antwort des endlichen Automaten aus Abbildung 2-3 [FOG00]

Ausgangszustand	C	B	C	A	A	B
Eingabe	0	1	1	1	0	1
Neuer Zustand	B	C	A	A	B	C
Ausgabe	β	α	γ	β	β	α

Der Optimierungsprozess der evolutionären Programmierung beginnt mit der Erzeugung einer Initialpopulation von endlichen Automaten. Die Fitness eines jeden endlichen Automaten wird gemessen anhand seines Voraussagevermögens innerhalb einer Problemumgebung. Zur Erlangung weiterer Generationen werden die besten Individuen einer Population selektiert und durch Mutation in Nachkommen überführt. Jedes Individuum erzeugt hierbei genau einen Nachkommen, es erfolgt keine Rekombination.

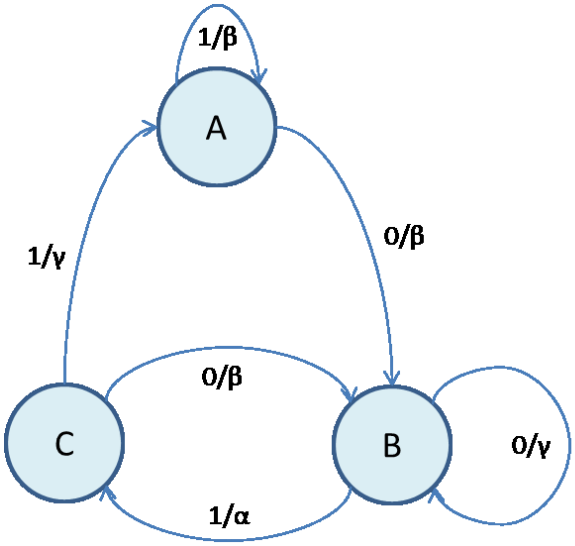


Abbildung 2-3: Endlicher Automat mit drei Zuständen, nach [FOG00]

Die Mutation kann folgende Veränderungen an den Individuen vornehmen:

- Änderung eines Ausgabesymbols
- Änderung einer Übergangsbedingung
- Hinzufügen eines Zustands
- Löschen eines Zustands

Die evolutionäre Programmierung ist zur Lösung einfacher Problemstellungen geeignet, stößt jedoch bei komplexeren Optimierungsaufgaben relativ schnell an ihre Grenzen [SOL66].

2.1.5 Genetische Algorithmen

Genetische Algorithmen wurden 1975 durch J. Holland [HOL75] eingeführt. Im Wesentlichen unterscheiden sich genetische Algorithmen durch drei Merkmale von den zuvor beschriebenen Evolutionären Algorithmen:

1. Die Repräsentationsform: ursprünglich Binärstrings
2. Die Selektionsmethode: Fitness-proportionale Selektion
3. Der primäre Optimierungs-Operator: Rekombination

Der von Holland vorgeschlagene Ansatz der Genetischen Algorithmen verwendet eine lineare, binär codierte Repräsentationsform. Grundsätzlich sind auch andere Formen der Repräsentation möglich, die jedoch ebenso die Einführung darauf abgestimmter Operatoren für Rekombination und Mutation erfordern. Die von Koza in [KOZ92] propagierte genetische Programmierung (siehe Kap. 2.1.6) stellt beispielsweise einen Sonderfall Genetischer Algorithmen dar. Die Interpretation des Binärstrings ist vollständig der Fitnessfunktion überlassen, der genetische Algorithmus bleibt hiervon unberührt.

Die Repräsentation numerischer Werte durch übliche Stellenwertsysteme, wie das Binärsystem oder auch das Dezimalsystem, ist für genetische Optimierung ungeeignet, da hier die Stärke einer Veränderung innerhalb der genetischen Information in keiner direkten Relation zur Stäke der erzeugten Auswirkung steht. Als Codierungsform hat sich, neben der Binär-Codierung, die Verwendung der Gray-Codierung als vorteilhaft erwiesen. Diese ist eine Form der binären Codierung, die sicherstellt, dass sich benachbarte, ganzzahlige Werte in der Gray-Codierung jeweils nur um ein Bit unterscheiden. Bei binärer Repräsentation treten oftmals signifikante Sprünge auf, wie in Tabelle 2-2 ersichtlich. Dadurch kann die Anwendung genetischer Operationen unerwünschte Nebeneffekte erzeugen.

Tabelle 2-2: Repräsentation ganzzahliger Werte in Binär- und Gray-Codierung

Dezimal-codiert	Binär-codiert	Gray-codiert
126	01111110	01000001
127	01111111	01000000
128	10000000	11000000

Der Prozess der Optimierung beginnt bei genetischen Algorithmen durch zufällige Erzeugung einer Initialpopulation mit einer festgelegten Anzahl an Individuen, entsprechend der vordefinierten Populationsgröße. Jedes der Individuen wird durch die Fitnessfunktion bewertet. Eine bestimmte Anzahl von Individuen wird anhand ihrer Fitness selektiert und in einen Nachkommen-Puffer kopiert. Hollands ursprünglicher Ansatz verwendete zur Selektion die Rouletterad-Selektion (siehe Kap. 2.1.2.2). Hierdurch wird sichergestellt, dass sich diejenigen Individuen mit einer höheren Fitness auch mit entsprechend höherer Wahrscheinlichkeit vermehren als die Individuen mit geringerer Fitness.

Die Individuen des Nachkommen-Puffers werden durch Anwendung der genetischen Operationen – Mutation und Rekombination - variiert. Ob für ein jeweiliges Individuum eine Mutation oder eine Rekombination stattfindet, wird anhand globaler Mutations- bzw. Rekombinationsraten entschieden.

2.1.5.1 Rekombination

Im Fall binär codierter oder gray codierter Repräsentation erfolgt die Rekombination zweier Eltern-Individuen durch gegenseitigen Austausch eines Teiles der Erbinformation. Hierzu wird ab einer zufälligen Stelle innerhalb des Binärcodes die Information des jeweils anderen Elternteils übernommen. Hierbei handelt es sich um das *Single-Point-Crossover*. Meist werden zwei oder mehrere derartige Übergangspunkte ge-

wählt, so dass nur die genetische Information zwischen diesen Punkten ausgetauscht wird. In diesem Fall handelt es sich dann um das *Multi-Point-Crossover*, wie in Abbildung 2-4 mit zwei Übergängen dargestellt. Das Multi-Point-Crossover lässt sich prinzipiell auf alle linearen Repräsentationsformen, beispielsweise reelle Codierung, übertragen.

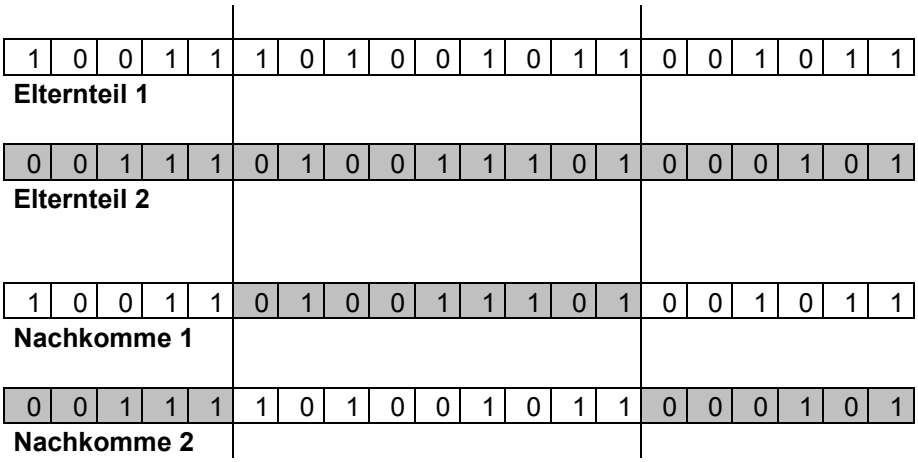


Abbildung 2-4: Multi-Point-Crossover im Fall binärer Repräsentation

Diese Art der Rekombination bewirkt oftmals unerwünschte Seiteneffekte. So wird zu keinem Zeitpunkt Rücksicht auf den Kontext der Informationen genommen, die durch die binär codierte genetische Information repräsentiert wird. Oftmals erfolgt so die Trennung wichtiger zusammenhängender Einheiten, die nach der Rekombination keinen Sinn mehr ergeben. Die Rekombination kann beispielsweise einen Zahlenwert derart trennen, dass die höherwertigen Bits aus Elternteil 1 und die geringerwertigen aus Elternteil 2 zusammengesetzt werden, was einen eher zufälligen Zahlenwert ergibt, der sich höchstwahrscheinlich in beiden Nachkommen ungünstig auswirkt. Dieser Problematik lässt sich durch eine besser geeignete Repräsentationsform begegnen, beispielsweise reell- oder baumförmige Codierung.

2.1.5.2 Mutation

Für die Mutation binär codierter Individuen existieren mehrere Verfahren, die stark von der Art der repräsentierten Information abhängt. Üblicherweise kommt die sog. Flip Mutation (Abbildung 2-5) zum Einsatz. Hierbei erfolgt die Invertierung eines einzelnen Bits an beliebiger Stelle. Sofern es sich um binäre Codierung handelt, können durch die Negierung eines einzelnen Bits in Abhängigkeit von dessen Position gleichwohl geringe wie starke Veränderungen des Individuums erfolgen. Letztere treten beispielsweise bei Manipulation eines hochwertigen Bits eines numerischen Wertes auf, der durch eine binäre Sequenz innerhalb des Individuums repräsentiert wird. Besonders in diesem Fall hat sich die Gray-Codierung (siehe Tabelle 2-2) als vorteilhaft erwiesen, da die Änderung eines Bits hier auch immer nur eine geringe Veränderung des numerischen Wertes bewirkt.

Eine weitere Form der Mutation, die bei kombinatorischen Problemen sinnvoll sein kann, ist die *Insert-Mutation* [POH99] (Abbildung 2-6). Hier wird ein beliebiges Bit aus der Binärfolge entnommen und an anderer Stelle eingefügt. Die verbleibenden Bits

rücken entsprechend nach. Weiterhin existiert eine Reihe ähnlicher Mutationsoperatoren, die einzelne Bits oder ganze Bitgruppen in ihrer Anordnung variieren.

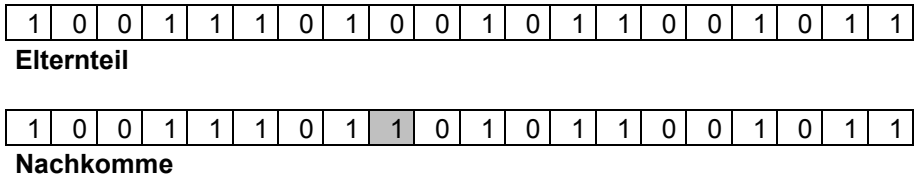


Abbildung 2-5: Mutation einzelner Bits bei binärer Codierung

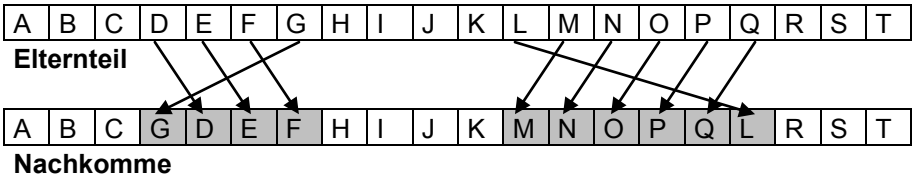


Abbildung 2-6: Insert-Mutation

Im Fall reell codierter Individuen kommen üblicherweise andere Mutationsoperatoren zum Einsatz. Die kleinsten Einheiten der reell codierten Information sind in sich geschlossene, numerische Werte. Die gewünschten geringfügigen Veränderungen dieser Informationen werden durch Addition bzw. Subtraktion geringer Anteile zu den ursprünglichen Werten erreicht. Der Definitionsbereich der zu addierenden Anteile sowie deren eigentliche Bestimmung haben einen großen Einfluss auf das Optimierungsverhalten. Große Werte ermöglichen eine relativ schnelle Abdeckung des Suchraumes, während geringere Werte eine genauere Optimierung gefundener Extrembereiche ermöglichen. Durch variable Steuerung der Optimierungsparameter während des Optimierungsverlaufs wird die positive Ausnutzung beider Verhaltensweisen ermöglicht.

2.1.5.3 Schema-Theorem

Das Schema-Theorem nach J. Holland [HOL75] wird oft als eine der wichtigsten Grundlagen zur Erklärung der Frage angesehen, warum und wie genetische Algorithmen funktionieren. Ein Schema ist eine Untermenge der genetischen Information eines Problemraums, die an bestimmten Stellen des genetischen Codes definierte Merkmale aufweist.

Schemata können bei binärer Repräsentation beispielsweise in der Form 1****1***0** notiert werden. Der Stern ist hierbei ein Platzhalter für beliebige Werte. Sämtliche Individuen, die diesem Muster entsprechen, können dem entsprechenden Schema zugeordnet werden. Gleichzeitig kann jedes Individuum mehreren Schemata angehören.

Ein Schema wird nach zwei wesentlichen Kenngrößen klassifiziert: Der Ordnung sowie der Definitionslänge. Die Ordnung gibt die Anzahl der durch das Schema festgelegten Bits an. Die Definitionslänge ist der größte Abstand der festgelegten Bits. Das obige Beispiel hätte eine Ordnung von 3 und eine Definitionslänge von 9. Die Fitness eines Schemas lässt sich durch Bildung des Mittelwertes der Fitness aller

möglichen zugehörigen Individuen bewerten, im obigen Fall sind dies 2^9 = 512 Individuen.

Das Schema-Theorem nach Holland besagt, dass Schemata geringer Ordnung mit überdurchschnittlicher Fitness in erfolgreichen Generationen exponentiell ansteigen. Mit „ansteigen“ ist hierbei die Anzahl der einem Schema zugehörigen Individuen gemeint. In (2-10) ist dies mathematisch ausgedrückt [POH99]. Hierbei sind *m* die Anzahl der zu einem Schema *h* gehörigen Individuen, $F_{(h)}$ ist die Fitness des Schemas und A_t bezeichnet die durchschnittliche Fitness der gesamten Population. Die Wahrscheinlichkeit *p* der Zerstörung des Schemas durch Mutation oder Rekombination ist in (2-11) gegeben. Die Definitionslänge $\delta_{(H)}$ des Schemas im Verhältnis zur Länge l der Individuen bildet zusammen mit der Wahrscheinlichkeit der Rekombination p_c das Risiko der Zerstörung durch Rekombination. Das Zerstörungsrisiko infolge Mutation ergibt sich aus der Ordnung $o_{(H)}$ des Schemas multipliziert mit der Wahrscheinlichkeit der Mutation p_m.

Schema-Zugehörigkeit:

$$m_{(h,t+1)} \geq \frac{m_{(h,t)} \cdot F_{(h)}}{A_t} \cdot (1-p) \qquad \textbf{(2-10)}$$

Zerstörungs-Wahrscheinlichkeit:

$$p = \frac{\delta_{(H)}}{l-1} \cdot p_c + o_{(H)} \cdot p_m \qquad \textbf{(2-11)}$$

2.1.6 Genetische Programmierung

Eine Sonderform genetischer Algorithmen ist die genetische Programmierung. Das Prinzip der genetischen Programmierung wurde erstmals umfassend durch John Koza in [KOZ92] beschrieben. Koza geht davon aus, dass sich fast alle Lösungen zu Optimierungsproblemen in Form von Computerprogrammen repräsentieren lassen. Ein Programm entspricht in diesem Zusammenhang einem Individuum einer Optimierungsaufgabe. Die Zielfunktion bewertet die Eignung dieses Programms zur Lösung der gestellten Aufgabe. Somit ist die genetische Programmierung zunächst prinzipiell eine Methode, die zur automatischen Erstellung von einfachen Computerprogrammen geeignet ist. Koza verwendete zur Repräsentation der Individuen die syntaktisch einfache Programmiersprache LISP.

2.1.6.1 Genotyp-Repräsentation

Die auf LISP basierenden Programme sind lediglich Phänotypen, deren Genotypen mittels einer syntaktischen Baumstruktur repräsentiert werden. Die Baumstruktur ist das wesentliche Kennzeichen der Genetischen Programmierung. Im Vergleich mit linearen Repräsentationsformen bietet sie etliche Vorteile bei Anwendung genetischer Operationen. Sie ist bei Weitem nicht auf die Repräsentation von LISP-Programmen oder syntaktischen Konstrukten beschränkt, sondern kann beliebige Datenstrukturen abbilden. Die im Folgenden dargestellten Beispiele sind daher unabhängig von der Programmiersprache LISP.

Eine Baumstruktur der genetischen Programmierung besteht aus terminierten und nicht terminierten Knoten. Nicht terminierte Knoten besitzen jeweils ein oder mehrere weitere Knoten, die ihrerseits terminiert oder nicht terminiert sein können. Beide Knotentypen folgen einem vorab definierten Vokabular, das die möglichen Symbole definiert. Die in Abbildung 2-7 dargestellte Baumstruktur einer mathematischen Funk-

tion erlaubt für nicht terminierte Knoten die Symbole „+, -, *, /“ und für terminierte Knoten die Symbole „1, 2, 3, 4, 5, 6, 7, 8, 9, x“.

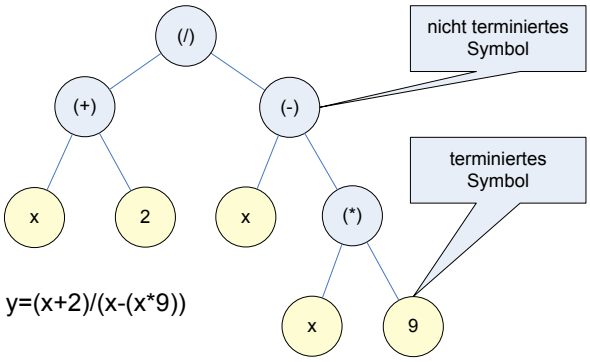


Abbildung 2-7: Baum-Struktur eines Individuums

Phänotyp:

$$y = \frac{x+2}{x - x*9} \qquad \textbf{(2-12)}$$

2.1.6.2 Rekombination

Die für lineare Repräsentationsformen üblichen Rekombinations- und Mutationsoperatoren sind auf Individuen, die mittels syntaktischer Bäume repräsentiert werden, nicht sinnvoll anwendbar.

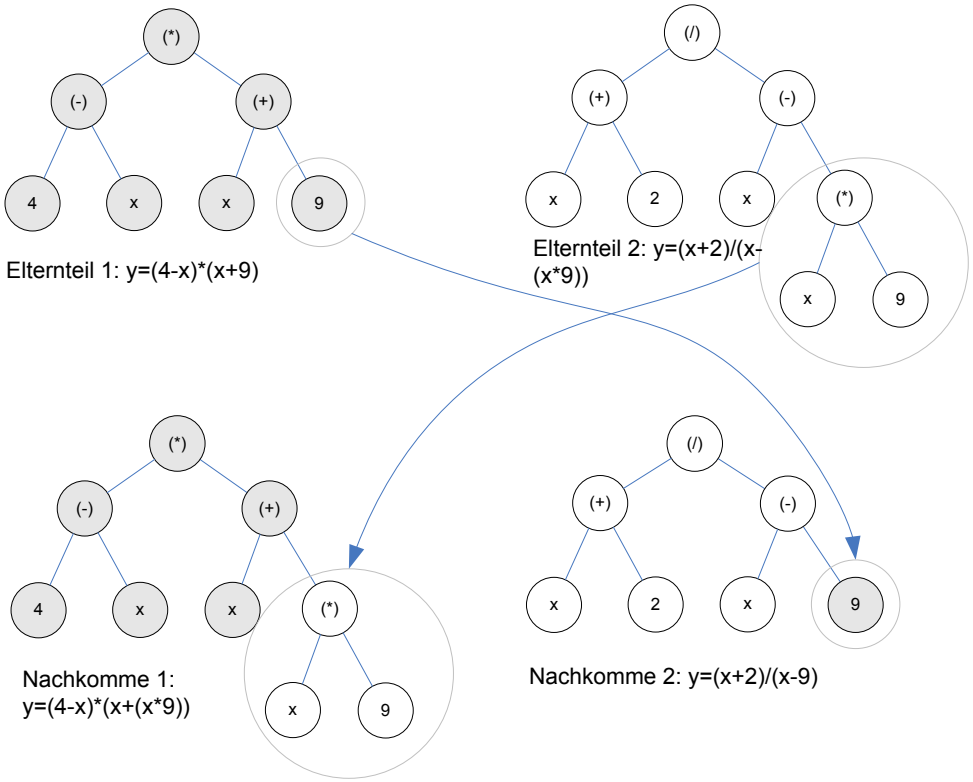


Abbildung 2-8: Point-Crossover

Die Rekombination zweier Individuen erfolgt bei genetischer Programmierung durch Austausch von Teilbäumen ab bestimmten Knoten. Üblicherweise können hier zwei beliebige Knoten gewählt werden. Dies führt jedoch schnell zu Phänomenen wie beispielsweise Bloat (siehe Kap. 4.2.7).

In [LAN02] wird daher das *Point-Crossover* vorgeschlagen. Hierbei erfolgt zunächst das *Aligning*, eine Identifikation der Form der beiden Elternbäume nach möglichen gemeinsamen Kreuzungsknoten. In beiden Elternbäumen wird jeweils ein Knoten an geeigneter Position für die Kreuzungsoperation ausgewählt. Die unter den gewählten Knoten befindlichen Teilbäume werden vollständig gegeneinander ausgetauscht.

2.1.6.3 Mutation

Für die genetische Programmierung kommen unterschiedliche Mutationsoperatoren zum Einsatz. Die bekannteste Variante der Mutation bewirkt die vollständige Zerstörung und anschließende Reinitialisierung von Teilbäumen beginnend an einem selektierten Knoten (Abbildung 2-9). Die an der Stelle des ursprünglichen Teilbaums entstehende Lücke wird durch einen zufällig initialisierten, jedoch synthaktisch korrekten Teilbaum ersetzt. Die möglichen Auswirkungen und der Grad der Veränderung an der genetischen Information des Individuums hängen hierbei entscheidend von der Tiefe des selektierten Knotens innerhalb der Baumstruktur ab.

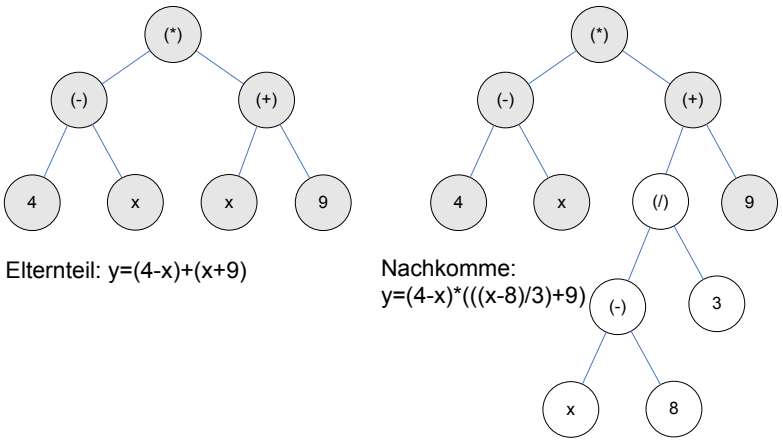


Abbildung 2-9: Reguläre GP-Mutation

Die Mutation eines Knotens auf unterer Ebene bewirkt geringfügige Änderungen, ein höher verankerter Knoten hingegen kann einen bedeutenden Teil des Individuums abändern. Hierdurch entstehen ähnliche Probleme wie bei binärer Repräsentation, wenn hochwertige Bits einer numerischen Repräsentation verändert werden und somit ungewollt starke Veränderungen des Individuums hervorrufen. Im Fall der genetischen Programmierung kann möglicherweise sogar das gesamte Individuum ausgetauscht werden, wenn der oberste Knoten zur Mutation selektiert wird.

Eine weitere Variante der Mutation ist die Point-Mutation (Abbildung 2-10). Im Unterschied zur regulären Mutation wird lediglich der selektierte Knoten selbst durch ein entsprechend gültiges Symbol ausgetauscht. Die anhängenden Knoten bleiben unverändert. Auch hier können unterschiedliche Grade der Veränderung auftreten, je nach Aussage der verwendeten Symbole. Die Anfälligkeit für weit reichende Reinitialisierungen entfällt jedoch in diesem Fall.

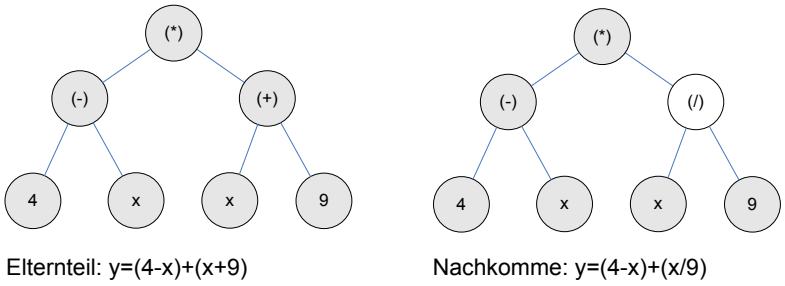


Abbildung 2-10: Point-Mutation

2.1.7 Anwendung im Rahmen dieser Arbeit

Für das in der vorliegenden Arbeit erforderliche Optimierungskonzept werden Verfahren benötigt, die zur Optimierung sowohl konzeptueller als auch detaillierter Tragwerksparameter erfolgreich eingesetzt werden können. Konzeptuelle Informationen folgen in großen Teilen einer hierarchischen Struktur, die je nach Ausprägung einen variablen Umfang annehmen kann. Daher erscheinen lineare Repräsentationsformen hier ungeeignet [DEB95]. Die Repräsentation konzeptueller Informationen lässt sich hingegen hervorragend in Form einer Baumstruktur realisieren. Hierbei kommt die genetische Programmierung zur Anwendung. So ist es möglich, sehr heterogene Informationen durch Separation der Teilbäume konsistent zu halten, während die Codelänge bei genetischer Programmierung unerheblich ist.

Parameter zur Repräsentation detaillierter Tragwerksinformationen sind meist reelle oder hinreichend fein diskretisierte numerische Werte. Deren Optimierung ist mit Hilfe einer reinen Baumstruktur nur bedingt möglich. Bessere Ergebnisse lassen sich durch Anwendung reell codierter genetischer Algorithmen erwarten. Im Rahmen der vorliegenden Arbeit wurde aus diesem Grund ein kombinierter Ansatz aus genetischer Programmierung und reell codierten Genetischen Algorithmen entwickelt.

2.2 Wissensbasierte Systeme

"Ein Experte ist ein Mann, der hinterher
genau sagen kann, warum seine Prognose nicht gestimmt hat."

Winston Churchill, 1874 - 1965

Wissensbasierte Systeme sind ein grundlegendes Konzept der künstlichen Intelligenz. Im allgemeinen Sprachgebrauch werden wissensbasierte Systeme oftmals mit Expertensystemen gleichgesetzt, wobei diese Analogie nicht zutreffend ist [BEI06]. Eine grundlegende Eigenschaft wissensbasierter Systeme ist die Bereitstellung von generischem Wissen, das zur Lösung einer abgegrenzten Problemdomäne erforderlich ist [BIB93].

Das allgemeingültige Domänenwissen wird oftmals auch als „Regelwissen“ bezeichnet, da es zu großen Teilen durch Regeln beschrieben wird. Gleichzeitig existiert Faktenwissen zu einer konkreten Problemstellung aus der Problemdomäne. Es erfolgt eine klare Trennung zwischen dem generischen Wissen in Form einer *Wissensbasis,* dem Faktenwissen (meist in Form eines Produktmodells) und dem zur An-

wendung des Wissens erforderlichen Auswertungsmechanismus, der *Inferenzkomponente*.

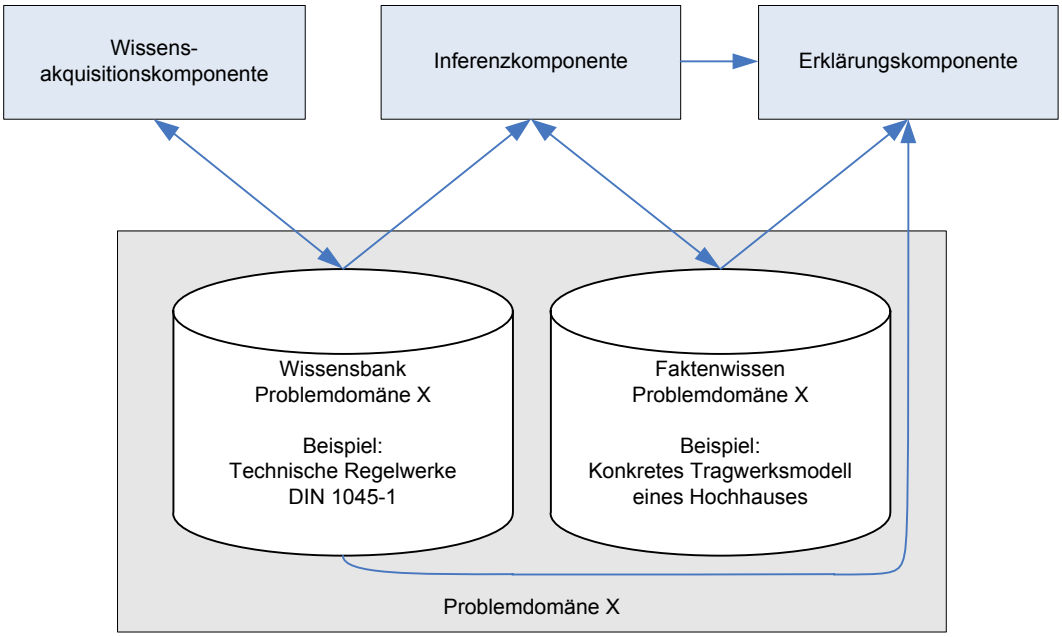


Abbildung 2-11: Grundlegender Aufbau eines wissensbasierten Systems

Expertensysteme spezifizieren diese Eigenschaften dahingehend, dass die Anwendung des Wissens zu ähnlichen Ergebnissen führen soll wie die Befragung eines menschlichen Experten. Hierzu zählt neben der reinen Lösung eines Problems die detaillierte Nachvollziehbarkeit der Ergebnisse und des gesamten Lösungsweges. Expertensysteme beinhalten im Allgemeinen angepasste Mechanismen zur Schlussfolgerung und zur Verarbeitung ungenauen und unvollständigen Wissens. Somit stellen Expertensysteme lediglich eine Untermenge der wissensbasierten Systeme dar. Expertensysteme sind nicht zwangsläufig wissensbasierte Systeme, sie können auch auf anderen Technologien basieren, auch wenn dieser Fall unüblich ist.

2.2.1 Wissensrepräsentation

Die Wissensrepräsentation verfolgt das Ziel, Wissen durch Mathematisierung von Aspekten der Intelligenz formal abzubilden, so dass die computerbasierte Anwendung und Verarbeitung dieses Wissens ermöglicht wird. Die Art der Wissensrepräsentation ist primär abhängig von dem Einsatzzweck und den angewandten Verarbeitungs- und Schlussfolgerungsmechanismen. Ebenso hat die Form der Wissensrepräsentation einen starken Einfluss auf die Komplexität und Effizienz des Prozesses der Wissensakquisition.

Nach [WIK07/1] existieren folgende Systeme zur Wissensordnung und -repräsentation, aufsteigend nach ihrer Mächtigkeit:

1. Katalog, Taxonomie, Glossar
2. Klassifikation, Thesaurus
3. Semantisches Netz, Ontologie, Frames, Produktionsregeln

4. Axiomensystem, Prädikatenlogik
5. Mehrschichtige, erweiterte semantische Netze

Der im Rahmen der vorliegenden Arbeit verfolgte wissensbasierte Ansatz basiert auf der nicht-klassischen Prädikatenlogik höherer Ordnung [PAH00].

2.2.2 Wissensakquisition

Das Wissen, das einem wissensbasierten System zugrunde liegt, muss in einer für das jeweilige System verarbeitbaren Form bereitgestellt werden. Dieser Vorgang wird als „Wissensakquisition“ bezeichnet. Diese Aufgabe übernimmt eine mit der Problemdomäne vertraute Person, der „Wissensingenieur“. Da der Wissensingenieur in der Regel nicht über Programmierkenntnisse verfügt, ist ein allgemein verständlicher Formalismus entscheidend für die Möglichkeiten des Wissensingenieurs, sein Wissen sachgerecht und anwendbar auszudrücken. Ebenso ist die Bereitstellung entsprechender softwaretechnischer Werkzeuge zur komfortablen Wissensakquisition hilfreich. Dieser Umstand wird in der einschlägigen Literatur [BIB03], [BEI06], [GÖR95] weitgehend vernachlässigt, trägt jedoch entscheidend zur Leistung eines wissensbasierten Systems bei.

2.2.3 Inferenz

Die Inferenzkomponente eines wissensbasierten Systems stellt den eigentlichen Auswertungsmechanismus bereit, der zur Anwendung von allgemeingültigem Wissen auf problemspezifisches Faktenwissen erforderlich ist. Der Inferenzmechanismus, der dabei zur Anwendung kommt, ist abhängig von Art, Präzision und Vollständigkeit sowohl des generischen Wissens als auch des Faktenwissens.

Die drei wichtigsten Formen der Inferenz sind nach C. Pierce [HAR96] Deduktion, Abduktion und Induktion. Die Deduktion beschreibt den Schluss aus einer allgemeinen Aussage auf einen speziellen Sachverhalt. Sofern die zugrunde liegenden Aussagen korrekt sind, kann der Schluss einer Deduktion ebenfalls stets als korrekt angesehen werden. Die Induktion überträgt regelmäßig auftretende Sachverhalte auf eine allgemeingültige Anwendbarkeit. Das somit gewonnene Wissen ist womöglich für einen Großteil der auftretenden Fälle korrekt, jedoch nicht notwendigerweise für alle Fälle. Induktives Schließen erfolgt beispielsweise bei der Auswertung experimenteller Beobachtungen.

Derart gewonnene Regeln erscheinen plausibel und sind für die untersuchten Experimente korrekt, schließen aber nicht aus, dass Fälle existieren, für die diese Regeln keine richtige Aussage liefern. Im Fall der Induktion liegt somit unvollständiges oder ungesichertes Regelwissen vor. Unter Abduktion versteht Pierce die Schlussfolgerung basierend auf gesicherten Regeln, jedoch unvollständiger Fakten. Im alltäglichen Gebrauch werden die Fakten meist ergänzt durch Annahme eines Normalzustandes, der auf Erfahrung und regelmäßigen Erkenntnissen beruht. In der Wissensmodellierung wird dieser Normalzustand beispielsweise in Form von Default-Regeln abgedeckt [GÖR95].

Während die Deduktion aus gesichertem Wissen neues gesichertes Wissen schlussfolgert, basieren Abduktion und Induktion auf der Anwendung unvollständigen Wissens [BEI06]. Die Schlussfolgerung aufgrund unvollständigen Wissens ist unter dem Begriff „Nichtmonotones Schließen“ bekannt. Weitere Ansätze nichtmonotonen Schließens sind beispielsweise die Hinzunahme probabilistischer Informationen oder

die Übertragung analoger Sachverhalte unter optimistischer Annahme der Übertragbarkeit. Eine Übersicht gängiger inferenzieller Problemstellungen ist in Tabelle 2-3 gegeben.

Tabelle 2-3: Inferenzielle Problemstellungen, nach [BIB93]

Deduktion	W (vorhandenes Wissen) vollständig bekannt, B (neues Wissen) gesucht
Abduktion	W im faktischen Teil unvollständig
Induktion	W im Regelteil unvollständig

2.2.4 Erklärungskomponente

Ein wesentlicher Aspekt bei der Anwendung wissensbasierter Systeme ist die Sicherstellung der Transparenz. Dies beginnt bereits mit dem Vorgang der Erstellung und Anpassung einer Wissensbasis. Hierbei ist zu gewährleisten, dass die Wissensakquisitionskomponente das Wissen in geeigneter Form repräsentiert und den Anwender in die Lage versetzt, sein allgemeines Wissen in einer für ihn adäquaten Form auszudrücken. Ebenso ist jedoch auch der Anwender verantwortlich für eine strukturierte und nachvollziehbare Form der Wissensrepräsentation. Dies wird insbesondere bei Anwendung einer Wissensbank deutlich. Mittels einer Erklärungskomponente kann hierbei lückenlos nachvollzogen werden, auf welchen Annahmen und mittels welcher Mechanismen bestimmte Entscheidungen durch die Inferenzkomponente getroffen wurden.

2.2.5 Anwendung im Rahmen dieser Arbeit

Wissensbasierte Systeme spielen im Rahmen der vorliegenden Arbeit in zwei Bereichen eine bedeutende Rolle. Einerseits kommt ein wissensbasierter Ansatz zur Interpretation der Genotypen und letztendlicher Ausbildung der Phänotypen zum Einsatz. Dies hat eine Reihe von Vorteilen für die praktische Anwendung des Optimierungssystems. Dem Anwender obliegt durch Definition von Bildungsvorschrift und zugehöriger Wissensbank die präzise Abgrenzung zwischen fest definierten Parametern, funktional oder regelbasiert abhängigen Werten und zu optimierenden Größen. Damit erreicht die Bandbreite der optimierbaren Strukturen größtmögliche Flexibilität, von der vollständigen freien Optimierung eines Tragwerks bis hin zur reinen Optimierung einzelner Parameter, beispielsweise Stützweiten oder Stockwerkskonfigurationen.

Weiterhin erfolgt die schlussendliche Bewertung der Phänotypen und Bildung der Fitnessfunktionen mittels eines weiteren wissensbasierten Ansatzes, der auf derselben wissensbasierten Plattform aufsetzt. Hierdurch wird sichergestellt, dass die Kriterien der Optimierung jederzeit den tatsächlichen Anforderungen angepasst werden können, was bei starr programmierten Auswertungen fixer Kriterien kaum möglich wäre. Die in der Zielsetzung der vorliegenden Arbeit geforderte interdisziplinäre Betrachtungsweise kann beispielsweise durch unabhängige Wissensbanken je Fachdisziplin realisiert werden, die jeweils das zur Bewertung eines Bauwerks erforderliche generische Wissen einer einzelnen Fachdisziplin repräsentieren. Ebenso erlaubt die wissensbasierte Auswertung die bedarfsgerechte Berücksichtigung weiterer Optimierungskriterien, die sich aus verschiedensten Umständen während der Planungsphase ergeben können.

2.3 Hochhaus-Tragsysteme

Der im Rahmen dieser Arbeit vorgestellte Optimierungsansatz kommt beispielhaft für die Optimierung von Hochhäusern zur Anwendung. Im folgenden Kapitel wird ein kurzer Überblick über die dafür erforderlichen Entwurfsgrundlagen gegeben.

2.3.1 Bauweise

Hochhäuser bestehen im Wesentlichen aus einer Skelettkonstruktion, die durch aussteifende Elemente wie Wände oder biegesteife Rahmen aus Stützen und Balken stabilisiert wird. Das Aussteifungssystem eines Hochhauses ist von fundamentaler Bedeutung für den gesamten Tragwerksentwurf. Nur durch die gesamtheitliche Betrachtung eines Tragwerkes einschließlich des Aussteifungssystems lässt sich dessen Qualität, Tragwirkung und Wirtschaftlichkeit beurteilen.

Die Beispiele dieser Arbeit beschäftigen sich ausschließlich mit der Stahlbetonbauweise. Das erarbeitete Softwarekonzept ist jedoch auch auf die Stahlbauweise oder die Verbundbauweise anwendbar.

2.3.1.1 Stahlbauweise

Die älteste Bauweise zur Ausbildung von Hochhaustragwerken ist die Stahlbauweise [KAR05]. Stahl weist im Verhältnis zu seinem Eigengewicht eine hohe Tragfähigkeit auf. Dadurch wird die Ausbildung schlanker Querschnitte ermöglicht, was der verbleibenden Nutzfläche zugute kommt. Durch einen hohen Grad an industrieller Vorfertigung kann ein zügiger Baufortschritt erreicht werden. Das geringe Eigengewicht des Stahls wirkt sich zudem günstig auf das Gesamttragwerk bis hin zur Fundamentierung aus.

Als nachteilig kann der hohe Aufwand durch zusätzlich erforderliche Brandschutzmaßnahmen angesehen werden. Ebenso ungünstig sind die geringere Steifigkeit und höhere Schwingungsanfälligkeit reiner Stahlkonstruktionen. Der relativ zu anderen Baustoffen überproportional steigende Stahlpreis wirkt sich negativ auf die Wirtschaftlichkeit von Stahlkonstruktionen aus.

2.3.1.2 Stahlbetonbauweise

Die Kombination aus dem relativ günstigen Baustoff Beton mit dem teureren Stahl ermöglicht im Allgemeinen einen sehr wirtschaftlichen Resourceneinsatz in vielen Bereichen des Hoch- und Tiefbaus. Im Falle von Hochhäusern wirkt sich das hohe Konstruktionseigengewicht mit zunehmender Höhe ungünstig aus. Erst durch die Entwicklung höherfester Betone ab 1960 sowie durch Weiterentwicklungen in der Betonfördertechnik gewann die Stahlbetonbauweise für den Hochhausbau zunehmend an Bedeutung. Dies ermöglichte die Ausbildung schlanker Stützenquerschnitte, die mit den Abmessungen von brandgeschützten Stahlstützen vergleichbar sind.

Stahlbetonbauwerke weisen im Vergleich zu Stahlkonstruktionen ein günstiges Schwingungsverhalten auf, da sie über eine hohe Steifigkeit und positive Dämpfungseigenschaften verfügen. Der Brandschutz der tragenden Elemente wird in der Regel durch den Beton selbst sowie ggf. zusätzliche konstruktive Maßnahmen erbracht. Die Herstellung von Betonkonstruktionen erfordert einen höheren Aufwand auf der Baustelle als bei Stahlkonstruktionen, da die Vorgänge des Schalens und Bewehrens sehr zeitintensiv sind. Die Beachtung von einzuhaltenden Ausschalfristen erfordert zudem eine mehrfache Vorhaltung von Rüst- und Schalmaterialien und erzeugt aus baubetrieblicher Sicht komplexere Bauabläufe. In wirtschaftlicher Hinsicht überwiegen jedoch im Allgemeinen die Vorteile der Stahlbetonbauweise. In den

letzten Jahren wird zudem vermehrt auf Betonfertigteile (bzw. Halbfertigteile) zurückgegriffen, die zur Steigerung der Wirtschaftlichkeit beitragen.

2.3.1.3 Verbundbauweise

Neben der reinen Stahl- oder Stahlbetonkonstruktion kann der Einsatz von Verbundbauteilen sinnvoll sein. Verbundbauteile weisen eine hohe Tragfähigkeit auf. Vorgefertigte Stahlteile, die mit Ortbeton zu Verbundbauteilen ergänzt werden, vereinen die Vorteile industrieller Vorfertigung mit vermindertem Schalungsaufwand für den Beton, unter weitgehender Beibehaltung der Vorteile von Betonbauteilen im Hinblick auf Brandschutz, Steifigkeit und Schwingungsdämpfung.

2.3.2 Stahlbeton-Bauteile

2.3.2.1 Decken

Das Deckensystem stellt einen erheblichen Kostenfaktor bei Stahlbeton-Hochhäusern dar, da die Decken das größte Betonvolumen der Gesamtkonstruktion ausmachen. Je nach Belastung, Spannweite und Führung von Installationsleitungen können Flachdecken, Decken mit Unterzügen oder Rippendecken wirtschaftlicher sein. Bei punktgelagerten Flachdecken tritt oftmals das Problem des Durchstanzens an den Stützenauflagern auf, dem durch Anordnung von Durchstanzbewehrung sowie Verwendung hochfester Betone begegnet werden kann.

2.3.2.2 Stützen

Die primären vertikalen Tragglieder eines Hochhauses sind Stützen. Aus Gründen der Ästhetik werden hier in vielen Fällen neben großen Stützenabständen auch immer schlankere Querschnitte gefordert. Gleichzeitig sollen Stützen oft in großen Bereichen von Stockwerken gleichmäßige Querschnitte aufweisen. Daher erfolgt häufig der Einsatz unterschiedlicher Betonsorten je nach Geschoss. Die Verwendung von hochbelastbaren Verbundstützen galt lange als wirtschaftlich, da schlanke Querschnitte bei den Vorzügen der Materialeigenschaften von Beton ermöglicht wurden. Verbundstützen sind jedoch relativ kostenintensiv in der Herstellung. Daher werden sie zunehmend von Stützen aus hochfestem Beton verdrängt.

2.3.2.3 Wände

Wände dienen neben ihrer vertikalen Tragwirkung auch als aussteifende Elemente. Besonders häufig werden Wände im Verbund als zusammengesetzte Querschnitte ausgebildet, die neben der aussteifenden Wirkung zur vertikalen Erschließung des Gebäudes dienen, z.B. als Treppenhaus, Versorgungs- und Aufzugsschacht. Ebenso kommen Wände als aussteifende Fassadenelemente zum Einsatz, üblicherweise in Form von Lochfassaden. Im Hochhausbau werden die Wände außerhalb der Kernbereiche meist als Einschränkung in der Grundrissgestaltung angesehen, da aus Gründen der Variabilität bevorzugt flexibel gestaltbare Leichtbau-Trennwände eingesetzt werden.

2.3.2.4 Unterzüge

Stahlbeton-Unterzüge finden oftmals bei großen Deckenspannweiten Verwendung, da Flachdecken mit zunehmenden Spannweiten unwirtschaftlich werden und zu große Verformungen aufweisen. Teilweise erfolgt eine Kombination aus Haupt- und Nebenunterzügen. Bei biegesteifem Verbund mit Stützen oder Wänden können Unterzüge auch zur Unterstützung des Aussteifungssystems herangezogen werden,

wie beispielsweise bei den Türöffnungen in Aufzugsschächten. In diesem Fall werden die Unterzüge als „Riegel“ bezeichnet.

2.3.3 Aussteifungssysteme

2.3.3.1 Rahmen

Rahmenkonstruktionen stellen die älteste Aussteifungsart des Hochhausbaus dar. Durch biegesteife Verbindung von Stützen und Riegeln übernehmen diese durch Biegung den Abtrag auftretender Horizontalkräfte. Diese Bauweise bietet die meisten Freiheitsgrade in Bezug auf die Grundrissgestaltung, sie ist jedoch bedingt durch die geringe Steifigkeit nur bei niedrigen Bauhöhen oder in Kombination mit übergeordneten Tragstrukturen wirtschaftlich einsetzbar.

Abbildung 2-12 und 2-13: Commerzbank Hauptzentrale, Frankfurt am Main

Das Commerzbank-Hochhaus in Frankfurt am Main (Abb. 2-12 und 2-13) ist ein Beispiel für den erfolgreichen Einsatz von Rahmenkonstruktionen in Verbindung mit einer übergeordneten Megastruktur (siehe Kap. 2.3.3.7). Das ca. 300 m hohe Gebäude ist das erste in Stahlbeton-Verbundbauweise errichtete Gebäude dieser Größe in Deutschland. Es besitzt einen dreieckigen Grundriss mit innen liegendem, durchgehendem Atrium. Die Seitenflächen werden durch viergeschossige Gärten in unterschiedlichen Höhen aufgelockert.

Die Deckenkonstruktion besteht aus 13 cm starken Ortbetonplatten aus Leichtbeton, die auf Stahlprofilblechen betoniert wurden. Stahlverbundträger im Abstand von 3 m und einer Spannweite von 15,6 m leiten die Deckenlasten in die Fassadenkonstruktion. Die Fassade besteht aus biegesteifen Vierendeel-Rahmen, die sich größtenteils über acht Geschosse erstrecken. Die inneren Randträger auf der Seite des Atriums werden durch Verbundstützen in den Ecken getragen. Die außen liegenden Fassadenrahmen bilden in Verbindung mit den in den Ecken des Grundrisses befindlichen Mega-Stützen eine effiziente Röhrenkonstruktion [LAU04].

2.3.3.2 Fachwerksysteme

Die geringe Steifigkeit von Rahmen lässt sich durch zusätzliche diagonale Aussteifungen erhöhen. Fachwerke sind vorwiegend bei Stahlkonstruktionen üblich. Nachteilig sind diagonale Austeifungen im Hinblick auf die nutzbare Fläche und die Einschränkung der Grundrissgestaltung, da sie eine mit Wänden vergleichbare Konstruktionsfläche einnehmen. Der Einsatz von Fachwerksystemen kann in Verbindung mit Outrigger-Systemen (siehe Kap. 2.3.3.5) erfolgen, die eine Kopplung des Gebäudekerns an die Gebäudehülle ermöglichen.

2.3.3.3 Wandscheiben

Durch ihre Steifigkeit eignen sich Wandscheiben sehr gut zum Abtrag von Horizontallasten. Gleichzeitig übernehmen sie den vertikalen Abtrag von Normalkräften. Die Anordnung der Wände ist dabei so zu wählen, dass nicht alle Wände parallel zueinander sind und sich ihre Achsen nicht in einem gemeinsamen Punkt schneiden (Abbildung 2-14).

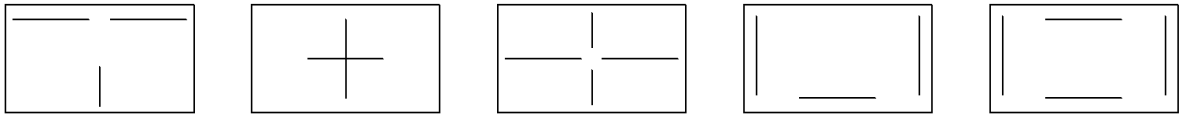


Abbildung 2-14: Anordnung von aussteifenden Wänden im Grundriss

2.3.3.4 Zusammengesetzte Querschnitte

Die monolithische Kopplung mehrerer Wandscheiben zu zusammengesetzten Querschnitten (meist Gebäudekernen) wirkt sich günstig auf die Stabilität aus. Dabei übernehmen Kerne meist auch die vertikale Erschließung des Gebäudes durch Aufzüge, Treppenhäuser und Installationsschächte. Neben statischen Gesichtspunkten ist dies auch in brandschutztechnischer Hinsicht von Vorteil.

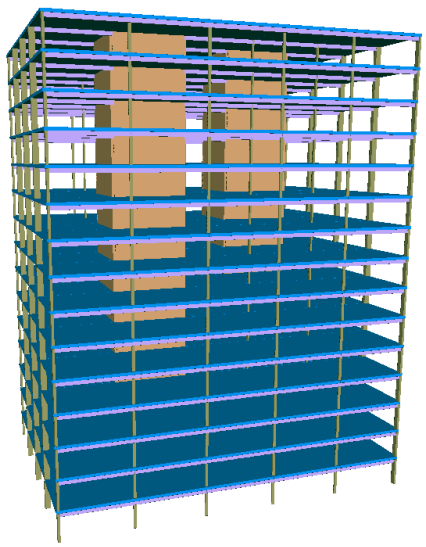


Abbildung 2-15: Aussteifung mit zwei Kernen

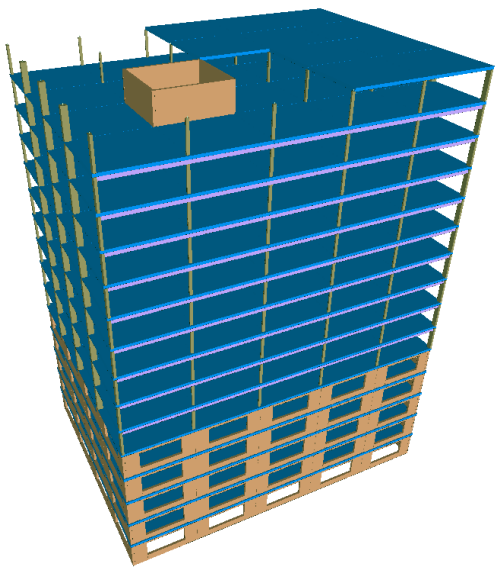


Abbildung 2-16: Kombinierte Aussteifung mit Kern und Lochfassade

Die Anordnung eines Kernes sollte möglichst zentral, oder bei mehreren Kernen symmetrisch erfolgen (Abbildung 2-15), um die durch Kerne aufzunehmenden Torsionsmomente infolge Ausmitte gering zu halten. Kerne können ebenso aus Rahmen oder Fachwerken gebildet werden, was im Stahlbetonbau jedoch unüblich ist.

2.3.3.5 Outrigger-Systeme

Outrigger ("Ausleger") sind koppelnde Scheiben- oder Fachwerkelemente, die Gebäudekerne schubfest mit den äußeren vertikalen Traggliedern in der Gebäudehülle verbinden. Durch Behinderung der Verdrehung des Kernes infolge von Horizontallasten werden diese teilweise durch die gekoppelten vertikalen Tragglieder aufgenommen. Hierbei unterscheidet man verschiedene Arten von Outrigger-Systemen.

Diskrete Outrigger-Systeme sind Fachwerkträger oder Wandscheiben, die in bestimmten Geschossen, meist Technikgeschossen, eine direkte Kopplung des Kernes an die Gebäudehülle bewirken. Nachteilig ist hierbei die Belegung eines Geschosses, das durch die innen liegenden Elemente nur bedingt nutzbar ist.

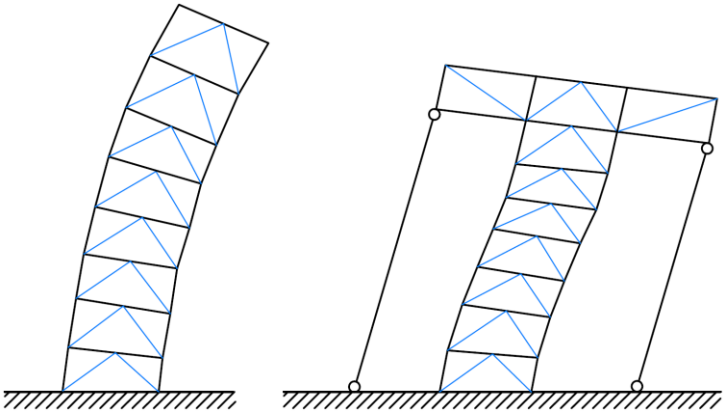


Abbildung 2-17: Unterschiedliche Verformung des Gebäudekerns ohne und mit diskretem Outrigger-System nach [PHO01]

Kontinuierliche Outrigger-Systeme entstehen durch Kopplung biegesteifer Deckenplatten an den Gebäudekern. Die stabilisierende Wirkung ist hierbei geringer als bei diskreten Outriger-Systemen, erlaubt jedoch eine durchgängige Nutzung aller Stockwerke.

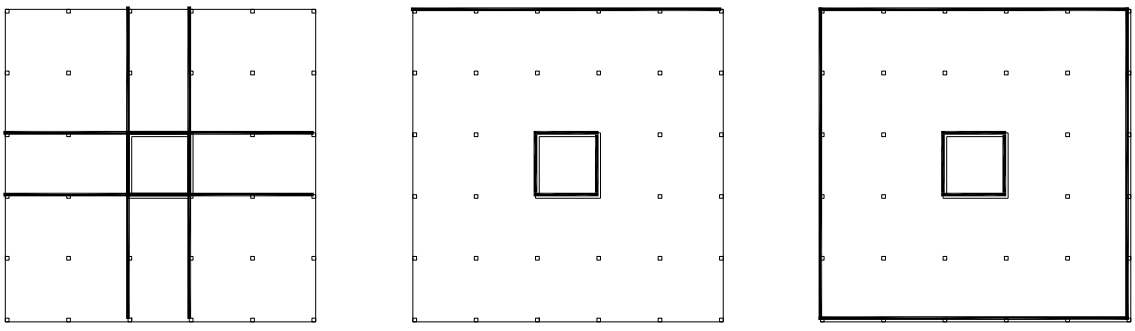


Abbildung 2-18: Outrigger-Systeme: Diskret, Offset und Belt-Truss

Die prinzipielle Wirkungsweise eines Outrigger-Systems ist in Abbildung 2-17 dargestellt. Der linke Teil zeigt die unbehinderte Verformung des Gebäudes infolge einer horizontal angreifenden Windlast. Auf der rechten Seite wird die Verdrehung des oberen Gebäudeteils durch einen diskreten Outrigger verhindert. Das dabei auftretende Moment wird durch ein Kräftepaar in der außen liegenden vertikalen Tragstruktur aufgenommen.

Eine indirekte Kopplung des Kernes wird durch außen liegende Offset-Outrigger ermöglicht. Diese verhindern die gegenseitige horizontale Verschiebung der anschließenden Deckenscheiben und bewirken somit eine Verformungsbehinderung des Kernes. Eine Sonderform der Offset-Outrigger sind Belt-Truss Systeme. Hierbei befinden sich in der Gebäudehülle der Outrigger-Geschosse vollständig umlaufende Wandscheiben. Daher ähnelt die Fassade optisch an diesen Stellen einem Gürtel.

2.3.3.6 Röhrenaussteifungen

Röhrensysteme entstehen durch Aktivierung der äußeren Gebäudehülle als aussteifendes Element. Dies erfolgt durch schubfeste Verbindung tragender Elemente wie Rahmen, Fachwerk- oder Fassadenelemente einer Lochfassade (Abbildung 2-16 und Abbildung 2-19) zu einem geschlossenen Hohlkasten.

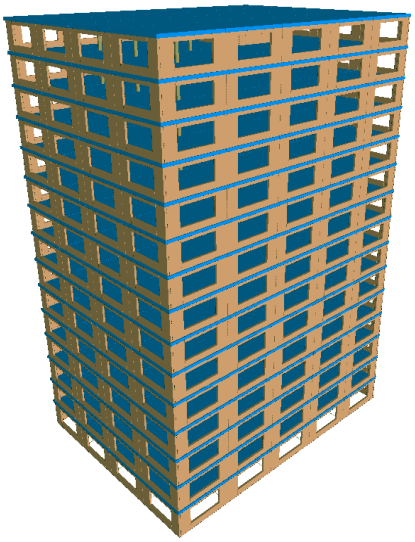


Abbildung 2-19: Lochfassade

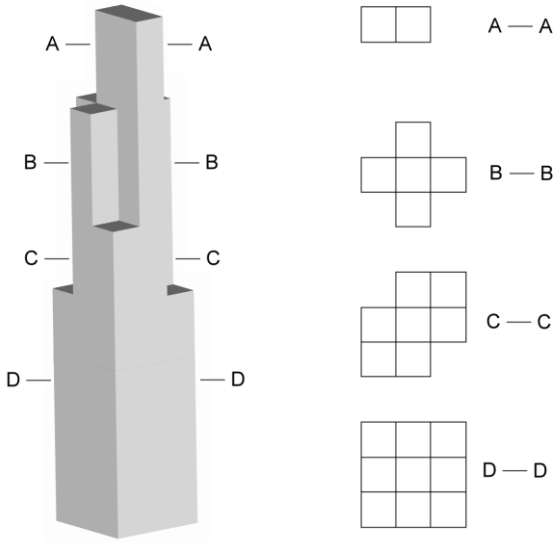


Abbildung 2-20: Gekoppeltes Röhrentragwerk; Sears-Tower, Chicago, nach [PHO01]

Röhrenaussteifungen sind eine sehr effektive Form der Gebäudeaussteifung und können zur Aussteifung von Hochhäusern mit einer Gebäudehöhe von über 200 m verwendet werden. Eine zusätzliche Steigerung der Aussteifungsleistung kann durch Kombination mehrerer Röhrentragwerke erreicht werden. Bei Rohr-in-Rohr Systemen erfolgt die Kopplung der äußeren Röhre mit einem darin befindlichen Gebäudekern.

Die Kopplung kann durch Deckenelemente oder besser durch Outrigger erfolgen, wodurch sich eine zusätzliche Rahmentragwirkung zwischen äußerer Röhre und Kern einstellt. Eine weitere Möglichkeit ist die Verbindung mehrerer paralleler Röhren zu einem System aus gebündelten Röhren, wie beispielsweise die Konstruktion des Sears Tower, Chicago (Abbildung 2-20).

2.3.3.7 Megastrukturen

Die Ausbildung einer übergeordneten mächtigen Tragstruktur wird als Megastruktur bezeichnet. Megastrukturen bestehen meist aus einer Kombination von Stützen und Riegeln zur Bildung von Mega-Rahmen und Mega-Fachwerksystemen. Typische Dimensionen eines Mega-Rahmens überspannen in der Vertikalen 10-15 Stockwerke. Megastrukturen können in einer Vielzahl von Varianten, von einfachen Megastüt-

zen bis hin zu komplexen gegliederten Mega-Raumfachwerken ausgebildet werden, wie am Beispiel des Shanghai World Financial Center in Abbildung 2-21 dargestellt. Zusätzliche Steifigkeit lässt sich durch Einsatz von Mega-Aussteifungselementen wie Diagonalverbänden erreichen (Abbildung 2-22).

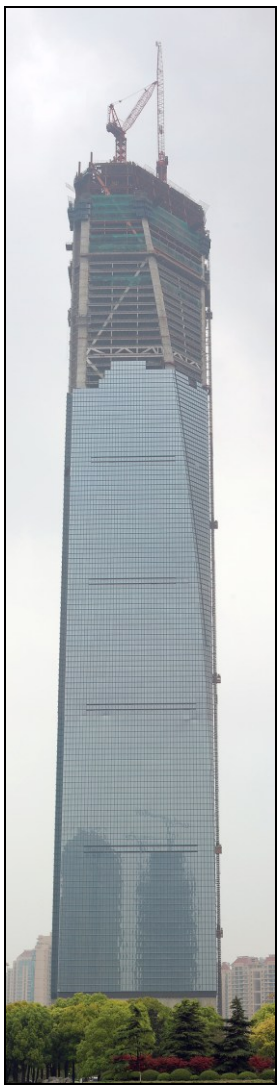


Abbildung 2-21: Shanghai World Financial Center, Bauzustand

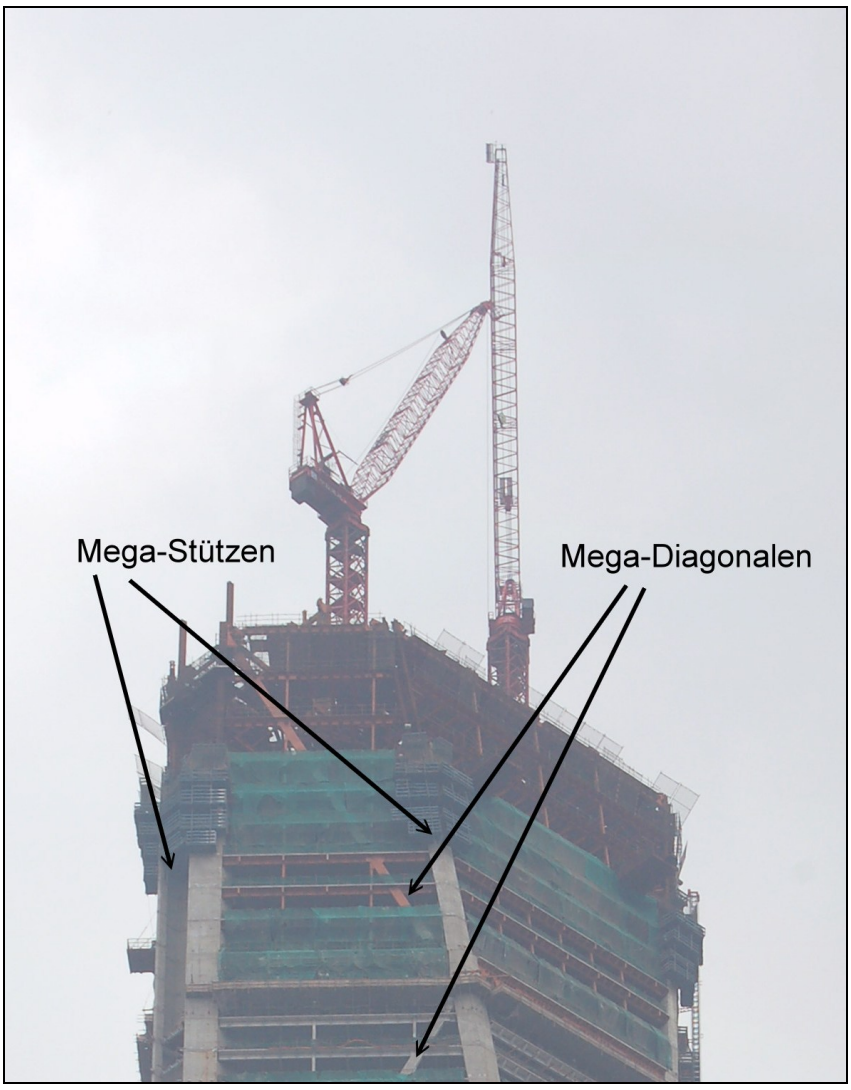


Abbildung 2-22: Mega-Raumfachwerk des Shanghai World Financial Center

Ein weiteres interessantes Beispiel für eine Megastruktur ist das Taipei 101 in Taiwan (Abbildung 2-23). Dieses Bauwerk wurde im Jahr 2004 fertig gestellt und ist derzeit eines der höchsten Hochhäuser weltweit. Da sich das Gebäude in einem Erdbebengebiet befindet, sind besonders hohe Anforderungen an die Konstruktion gegeben. Der Grundriss des Taipei 101 beinhaltet in der Gebäudehülle eine übergeordnete Tragstruktur aus acht Mega-Stützen, die durch Outrigger-Fachwerke mit dem Gebäudekern verbunden sind.

Die Mega-Stützen mit den Abmessungen 1*2 m [JOS06] bestehen aus einem geschlossenen Stahlprofil, das mit hochfestem Beton gefüllt ist. Die Schwingungsanfälligkeit des Bauwerks wird zusätzlich durch eine Tilgerkonstruktion reduziert (Abbildung 2-24). Diese besteht aus einer mehrteiligen, 726 t schweren Stahlkugel

[LIP06], die als Pendel zwischen dem 88. und 89. Stockwerk aufgehängt und durch Dämpfungsglieder mit dem Bauwerk verbunden ist.

Abbildung 2-23: Taipeh 101

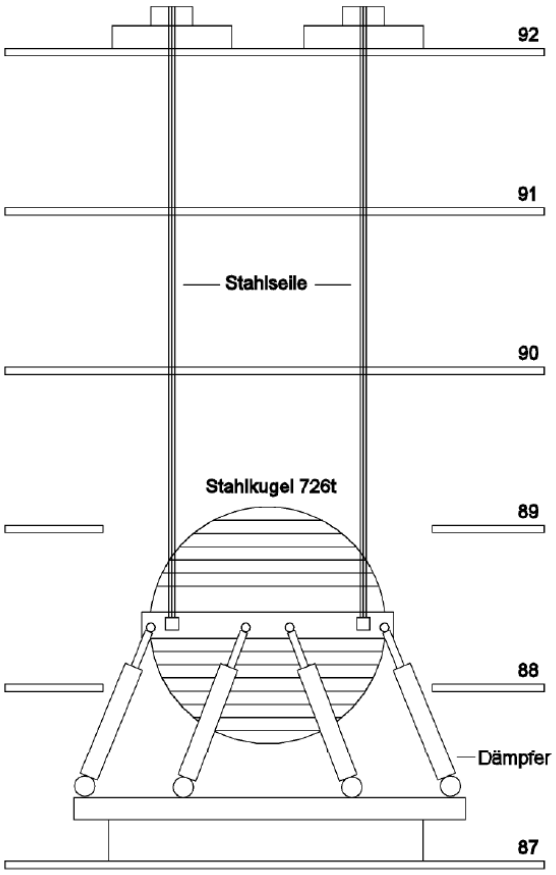


Abbildung 2-24: Tilgerkonstruktion des Taipeh 101

2.3.4 Anwendung im Rahmen dieser Arbeit

Der allgemeine Designprozess eines Hochhauses gliedert sich in zwei wesentliche Bereiche: Das architektonische Design sowie das Design des Tragwerks. Bei Hochhaustragwerken ist dies ein integrierter Prozess, da mit zunehmender Bauwerkshöhe die Tragstruktur stärkeren Einfluss auf die optische Gestaltung des Bauwerks nimmt. Somit entsteht eine hohe Abhängigkeit dieser beiden Disziplinen.

Die vorliegende Arbeit verfolgt das Ziel des Entwurfs einer Softwareplattform zur Optimierung von Stahlbeton-Tragwerken. Als Beispiel sollen Hochhaus-Tragwerke dienen. Die architektonische Gestaltung eines Bauwerks ist eine komplexe, durch vollständig automatisierte deterministische Verfahren nicht adäquat bewertbare Aufgabe. Die Optimierung des Tragwerks hingegen verfolgt in den meisten Fällen kontinuierlich bewertbare Ziele, beispielsweise den Entwurf einer möglichst kostengünstigen Konstruktion unter Beachtung der Auswirkungen auf weitere Fachdisziplinen. Gleichzeitig soll die maximale Nutzbarkeit des erstellten Bauwerks gewährleistet werden. Die Bewertung von Tragwerken hinsichtlich dieser Ziele lässt sich formal beschreiben, daher eignen sich Tragwerke prinzipiell für die Anwendung computergestützter Optimierungsverfahren. Im Rahmen der vorliegenden Arbeit wird von einem gegebenen architektonischen Design ausgegangen, dessen Tragwerk beispielhaft hinsichtlich Kosten und Nutzfläche optimiert werden soll.

Die automatisierte Unterstützung des Designs wirtschaftlicher Aussteifungssysteme mittels evolutionärer Optimierung lässt erwarten, dass sich die Tragwirkung mehrerer gekoppelter Systeme einstellt [ARC01]. Hierzu müssen dem System die entsprechenden Freiheitsgrade zur Erstellung und Kombination unterschiedlicher Tragsysteme zur Verfügung stehen. In umfangreichen Versuchsreihen in [PUL03] wurde gezeigt, dass sich geeignete Tragsysteme, wie beispielsweise zusammengesetzte Querschnitte, eigenständig während des Optimierungsprozesses herausbilden. Hierbei wurde jedoch lediglich ein einzelner gemeinsamer Grundriss über alle Stockwerke angenommen.

Die Übertragung des Optimierungsverhaltens auf dreidimensionale Tragwerke mit unterschiedlichen Ausprägungen je Stockwerk würde voraussetzen, dass sich im Laufe der Optimierung selbständig komplexe Verbände aus einzelnen Bauteilen zu mehrgeschossigen Kernen oder zu einem Belt-Truss System formieren. In diesem Fall müsste jedes einzelne Bauteil ein Teil der Beschreibung eines Individuums sein, was einen enorm großen Suchraum ergäbe und nicht mehr effizient optimiert werden könnte. Daher ist es notwendig, eine geeignete Repräsentationsform zu wählen, die die Definition grundlegender funktionaler Einheiten sinnvoller Tragsysteme und deren Kombination ermöglicht und dabei auf anerkannte und erprobte Grundmuster zurückgreift.

3 Gesamtkonzept

3.1 Zielsetzung

Im Rahmen der vorliegenden Arbeit wird ein flexibler, softwarebasierter Ansatz entwickelt, der zur Optimierung von Stahlbeton-Tragstrukturen geeignet ist. Besonderes Augenmerk liegt auf der Möglichkeit der freien Definition von Optimierungskriterien, Anpassbarkeit der gestalterischen und normativen Randbedingungen sowie der freien Skalierbarkeit der zu optimierenden Designparameter. Das System muss zudem in der Lage sein, die Belange unterschiedlicher Fachdisziplinen zu berücksichtigen.

3.2 Systemanalyse

3.2.1 Wissensrepräsentation

Für die praxisgerechte Anwendung des Optimierungsprozesses ist die interdisziplinäre Integration von Wissen unterschiedlicher Fachdisziplinen einer gemeinsamen Wissensdomäne erforderlich. Das fachspezifische Wissen ist strukturiert und transparent zu formalisieren. Grundsätzlich gilt, dass sich das für die Optimierung erforderliche Wissen einer jeden Fachdisziplin in folgende Bereiche unterteilen lässt:

1. **Allgemeines Entwurfswissen:** Dies ist allgemeines Wissen, welches zum Entwurf eines konkreten Simulationsmodells der betreffenden Fachdisziplin bei entsprechenden Vorgaben erforderlich ist.
2. **Allgemeines Bewertungswissen:** Dieses Wissen wird angewendet, um ein Simulationsmodell qualitativ sowie quantitativ zu bewerten (primär zur Ermittlung der Fitness).
3. **Spezifisches Wissen:** Die Erstellung eines Simulationsmodells einer Fachdisziplin erfordert detailliertes Wissen über die problemspezifischen Vorgaben und Randbedingungen. Einige dieser Variablen werden im Verlauf der Optimierung variiert, während andere konstant oder funktional abhängig sind.

Tabelle 3-1: Wissensarten und -repräsentationsformen

Wissensart	Einordnung	Beispiel	Repräsentationsform
Normwissen	Fachspezifisch	Wissen aus technischen Regelwerken, beispielsweise DIN-Normen (DIN 1045)	Wissensbank
Allgemeines Designwissen	Fachspezifisch	Domänenweit gültige Designregeln	Wissensbank
Spezifisches Designwissen	Fach- und Problemspezifisch	Für ein bestimmtes Optimierungsproblem spezifische Designregeln und Vorgaben	Wissensbank
Bewertungswissen	Fachspezifisch	Fitness-Bewertung, Kostenermittlung, etc.	Wissensbank
Optimierungsparameter	Problemspezifisch	Vorgabe der zu optimierenden Daten und deren Struktur	Bildungsvorschrift für Genotypen

Der angestrebte Grad an Flexibilisierung erfordert die grundlegende Trennung von domänen- und problemspezifischem Wissen von dessen softwaretechnischer Verarbeitung. Hierdurch wird zudem Transparenz und Nachvollziehbarkeit des Systemverhaltens sowie Wiederverwertbarkeit der allgemeinen Wissensanteile erreicht. Tabelle 3-1 zeigt eine Zusammenfassung der auftretenden Wissensarten nach Herkunft und Einordnung sowie mögliche Wissensrepräsentationsformen.

3.2.2 Bildungsvorschrift

Die zu optimierenden Variablen aller Fachdisziplinen bestimmen den strukturellen und inhaltlichen Aufbau der genotypischen Repräsentation eines Individuums in Form einer Bildungsvorschrift. Die Bildungsvorschrift enthält zusätzlich Informationen über mögliche Ausprägungen, Definitionsbereiche und Optimierungsparameter der einzelnen Variablen. Die Form der Bildungsvorschrift muss die adäquate Repräsentation der erforderlichen Freiheitsgrade der Optimierungsvariablen ermöglichen. Gleichzeitig sollten sämtliche der Bildungsvorschrift genügende Individuen in erster Instanz als gültig angesehen werden können.

3.2.3 Optimierungsvorgang

Der Optimierungsvorgang ist ein langwieriger, iterativer Prozess, der schrittweise zur Verbesserung der besten Individuen einer Population führen soll. Hierbei sind folgende Prozessschritte wiederholt zu durchlaufen:

1. **Bildung einer Initialpopulation:** Basierend auf der Bildungsvorschrift wird entsprechend der festgelegten Populationsgröße eine bestimmte Anzahl an Individuen zufällig oder zielgerichtet initialisiert. Bei externer Initialisierung wird anhand der Bildungsvorschrift verifiziert, ob es sich um ein gültiges Individuum handelt.
2. **Bildung einer Generation von Individuen:** Die in Schritten 1 oder 5 erzeugten Individuen bilden eine neue Generation innerhalb der betrachteten Population.
3. **Transformation der Genotypen in Phänotypen:** Sämtliche Individuen liegen innerhalb der aktuellen Generation in Form eines Genotypen vor. Unter Anwendung des formalisierten Entwurfswissens, des spezifischen Wissens sowie der genotypisch codierten Information wird für jedes Individuum ein bewertbares, interdisziplinäres Simulationsmodell erstellt. Dieses wird als Phänotyp bezeichnet.
4. **Simulation und Bewertung der Phänotypen:** Jeder Phänotyp wird anhand des wissensbasiert formalisierten Bewertungswissens sämtlicher betroffener Fachdisziplinen bewertet. Ein wesentlicher Teil des Bewertungswissens sind die Bewertungskriterien, die auf die jeweilige Optimierungssituation und deren Ziele individuell durch den Anwender angepasst werden können.
5. **Anwendung genetischer Operationen – Selektion, Kreuzung, Mutation**
6. **Ende der Optimierung bei Erreichen der Abbruchkriterien**

3.2.4 Übersicht

In Abbildung 3-1 sind die in der Systemanalyse identifizierten Wissensquellen sowie deren Anwendung im Rahmen des Optimierungsprozesses dargestellt. Die Wissensquellen sind gegliedert in allgemeines Wissen sowie Wissen, dass ein spezifisches Optimierungsproblem repräsentiert. Ebenso wurde die Aufteilung des Wissens in verschiedene Fachdisziplinen berücksichtigt.

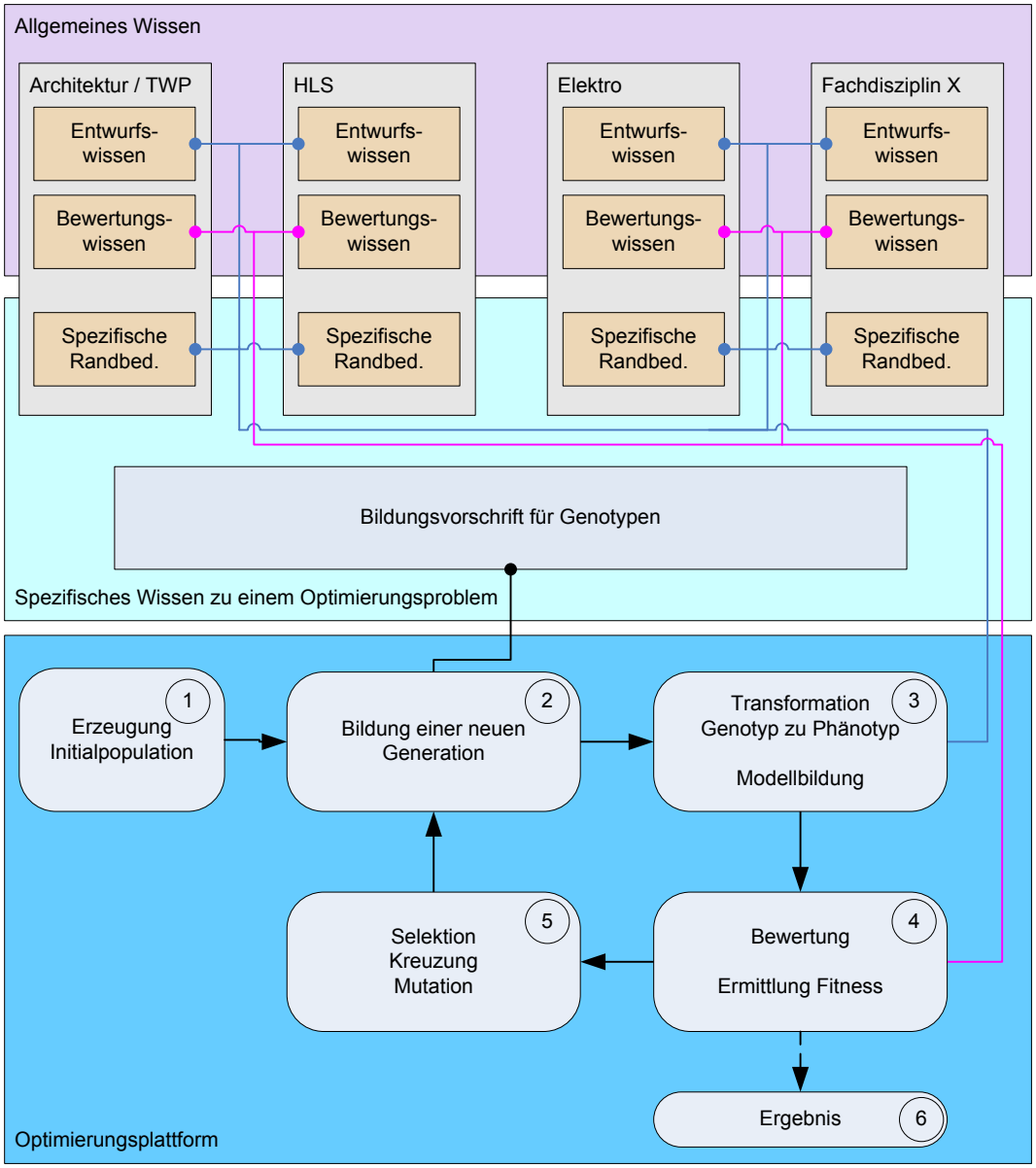


Abbildung 3-1: Übersicht der Systemanalyse

3.3 Systemkomponenten

3.3.1 Optimierungskomponente

Die Optimierungskomponente ist das zentrale Glied des Optimierungsprozesses. Sämtliche während einer Optimierung notwendigen Prozesse werden direkt oder indirekt durch die Optimierungskomponente gesteuert und kontrolliert.

Zur Sicherstellung größtmöglicher Flexibilität bei Definition von Optimierungsparametern wird zur Repräsentation der Bildungsvorschrift ein Grammatikbasierter Ansatz der genetischen Programmierung verwendet. Die Grammatik repräsentiert einen Satz von Produktionsregeln zur Erzeugung unterschiedlicher Individuen, die einer gemeinsamen Syntax unterliegen.

Die Definition und Wartung der Bildungsvorschrift ist eng gekoppelt mit dem Prozess der Wissensakquisition und sollte durch den Anwender erfolgen können. Die Bearbeitung der Bildungsvorschrift erfolgt mittels einer eigenständigen Akquisitionskomponente.

3.3.2 Wissensbasierte Plattform

Gemäß Tabelle 3-1 werden große Teile des domänen- und problemspezifischen Wissens durch einen wissensbasierten Ansatz erfasst und verarbeitet. Zu diesem Zweck wird eine komponentenbasierte wissensbasierte Plattform bereitgestellt.

Die Wissensakquisitionskomponente stellt die Benutzerschnittstelle zur Definition und Wartung der Wissensbanken dar. Die Inferenzkomponente dient der Anwendung des in Wissensbanken formalisierten Wissens unter Berücksichtigung der von der Optimierungskomponente bereitgestellten Parameter. Mit Hilfe einer Erklärungskomponente wird es dem Anwender ermöglicht, die Ergebnisse des Inferenzprozesses transparent nachzuvollziehen.

Das mit der wissensbasierten Plattform bereitgestellte Wissen deckt im Wesentlichen folgende Bereiche ab:

1. **Wissensbasierte Phänotyp-Erstellung:** Basierend auf den Vorgaben aus der genetischen Repräsentation eines Individuums sowie der Designregeln und Randbedingungen aus einer Design-Wissensbank erstellt die Inferenzkomponente ein dreidimensionales, parametrisiertes Bauwerksmodell in konzeptioneller und detaillierter Hinsicht, soweit dies für die Ermittlung der Fitness notwendig ist.
2. **Wissensbasierte Ermittlung der Einwirkungen:** Aus wissensbasiert formalisierten Normen ermittelt die Inferenzkomponente maßgebende Einwirkungen auf das Bauwerk (ständige Lasten, Verkehrslasten, Wind- und Erdbebenbelastung) als Vorbereitung der Schnittgrößenermittlung durch eine externe FE-Berechnungskomponente.
3. **Wissensbasierte Fitness-Ermittlung:** Basierend auf den Ergebnissen (Schnittgrößen) der externen Berechnungskomponente erfolgt eine wissensbasierte Bemessung sowie die Berechnung der Baukosten. Weitere durch den Anwender wissensbasiert vorzugebende Randbedingungen, wie erzielbare Erträge, statische oder topologische Eigenschaften, können ebenso ermittelt werden. Diese Informationen fließen gemeinsam in die Gesamtbewertung des erstellten Bauwerks in Form des Fitnesswertes.

3.3.3 FE-Berechnungskomponente

Zur hinreichend exakten Bewertung eines Tragwerks sind detaillierte Bemessungen der einzelnen tragenden Bauteile erforderlich. Grundlage der Bemessung ist die Ermittlung der Schnittgrößen unter allen maßgebenden Lastfallkombinationen. Ne-

ben den statischen Einwirkungen werden auch dynamische Einwirkungen infolge von Erdbeben- und Windlasten untersucht. Zur Ermittlung der Schnittgrößen kommt das an der Universität Duisburg-Essen entwickelte FE-System B&B [THI04] zum Einsatz.

3.3.4 Integriertes Entwurfssystem

Die beschriebenen Komponenten werden in einem gemeinsamen Entwurfssystem integriert. Das Entwurfssystem bildet die grafische Schnittstelle und die zentrale Bedieneinheit der Optimierungsplattform. Folgende zentrale Funktionen werden durch die Plattform bereitgestellt:

- Akquisition aller erforderlichen Wissensbanken
- Akquisition der Bildungsvorschrift
- Steuerung und Kontrolle des Optimierungsprozesses
- Verteilung der Rechenlast mittels einer eigenen Client-Server Architektur
- dreidimensionale Visualisierung einzelner Individuen während und nach der Optimierung
- detaillierte Nachvollziehbarkeit des wissensbasierten Tragwerksdesigns
- detaillierte Nachvollziehbarkeit der wissensbasierten Fitnesswertermittlung
- Export von Berechnungsergebnissen und Optimierungsverläufen

3.4 Systemarchitektur

Die in Abbildung 3-2 dargestellte Systemarchitektur gibt einen groben Überblick über die erforderlichen Komponenten. Es ist eine klare Trennung zwischen der genotypischen Repräsentation der zu optimierenden Individuen von deren phänotypischer Ausprägung zu erkennen. Die Bildung des Phänotypen sowie die Evaluation dessen Fitness erfolgt ausschließlich durch die Inferenzkomponente unter Anwendung der Informationen des Genotypen sowie des allgemeinen und spezifischen Problemlösungswissens mehrerer fachbezogener Wissensbanken. Diese enthalten unter anderem Regeln zur Interpretation der genotypischen Information.

Da der Phänotyp die vollständige Repräsentation eines Bauwerks darstellt, bedarf es zu dessen Vorhaltung ein relativ großes Datenvolumen. Anhand des Genotypen lässt sich der Phänotyp jederzeit wiederherstellen, daher wird dieser lediglich temporär während der Fitness-Evaluation vorgehalten.

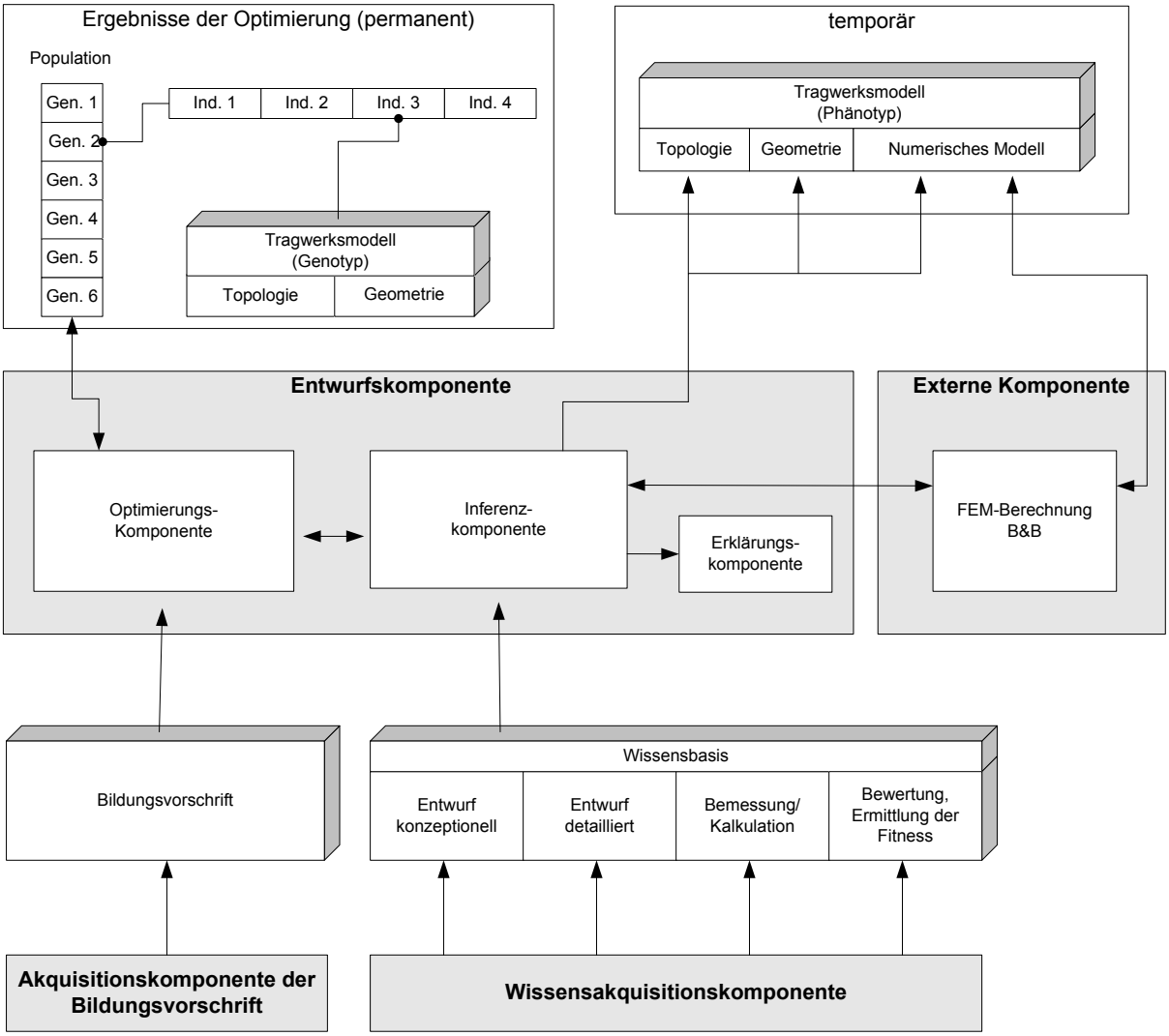


Abbildung 3-2: Systemarchitektur

4 Software-Design

Die im Rahmen der vorliegenden Arbeit entwickelte Softwareplattform umfasst Softwarekomponenten, die problemlos in weiteren Anwendungen zum Einsatz kommen können. Zur Sicherstellung der Wiederverwendbarkeit einzelner Module wurde beim Softwaredesign dieser Komponenten ein hoher Grad der Modularisierung berücksichtigt.

4.1 Wissensbasierte Plattform

Die im Rahmen dieser Arbeit entwickelte wissensbasierte Plattform basiert weitgehend auf den Arbeiten von Albert [ALB02], Garrett [GAR92] und Holéwik [HOL92]. Das von Albert entwickelte „OFWM“ (Objektorientiertes Fuzzy Wissensrepräsentationsmodell) ermöglicht die wissensbasierte Formalisierung von Entwurfs- und Bemessungswissen, angewendet auf Problemstellungen des Bauwesens. Im Rahmen der vorliegenden Arbeit wurde dieses Modell um zahlreiche Komponenten und Funktionen erweitert, um neben dem Bemessen und dem Führen von Nachweisen auch die wissensbasierte Erstellung von Bauwerksmodellen zu ermöglichen.

Primäres Ziel dieses wissensbasierten Systems ist die größtmögliche Anpassbarkeit des vorliegenden Wissens durch einen „Wissensingenieur“. Im Gegensatz zu klassischen Expertensystemen, die meist rein regelbasiertes Wissen verarbeiten [BAL01], liegt das Wissen hier in Form von Formeln, Tabellen, Diagrammen, Anwendungsregeln und Selektionen vor. Diese „Berechnungselemente“ ermöglichen die Repräsentation des Wissens in einer ähnlichen Form, wie sie auch in Normen und Regelwerken zu finden ist. Somit entsprechen sowohl die Wissensakquisition, als auch die spätere Nachverfolgung des angewendeten Wissens innerhalb einer Erklärungskomponente der gewohnten Arbeitsweise eines Ingenieurs.

4.1.1 Domänen-Unabhängigkeit

Das in Wissensbanken zu formalisierende Wissen bezieht sich immer auf eine Problemdomäne. Die Berechnungselemente einer Wissensbasis müssen mit dem Schema der Problemdomäne korrespondieren, um auf konkrete Problemstellungen der Domäne anwendbar zu sein. Ebenso ist es erforderlich, Schnittstellen zwischen dem Domänenwissen und der Wissensbank zu schaffen, um problemspezifische Informationen weiterverarbeiten zu können. Die Schnittstellen sind auf Seiten der Problemdomäne fest implementiert, stellen jedoch keinen Teil der wissensbasierten Plattform dar, sondern werden lediglich zum Auswertungszeitpunkt von dieser angewendet.

Um die Schnittstellen-Informationen der Wissensdomänen unabhängig von der Implementierung der wissensbasierten Plattform zu halten, wurde ein Datei-basiertes Schema-Austauschformat geschaffen. Somit wird ermöglicht, dass mehrere Software-Applikationen die Inferenzkomponente der wissensbasierten Plattform auf ihre spezifischen Problemdomänen hin anwenden können. Hierzu muss die jeweilige Software-Applikation in der Lage sein, das Schema ihrer zugrunde liegenden Datenstruktur in Form des spezifizierten Schema-Austauschformates zu exportieren.

Eine Software-Applikation kann hierbei mehrere Schemata für verschiedene Anwendungsfälle bereitstellen. In der vorliegenden Arbeit existieren beispielsweise die zwei Domänen *Phänotyp-Generierung* und *Fitness-Evaluation*. Abbildung 4-1 zeigt beispielhaft die Verwendung zweier unterschiedlicher Problemdomänen in Verbindung mit dem wissensbasierten System.

Das Schema-Austauschformat enthält sowohl die hierarchische Struktur einer Problemdomäne, als auch Informationen über die von der Domäne bereitgestellten Schnittstellen und Parameter. Die Anwendung der Schnittstellen erfolgt im Rahmen des Berechnungselementes „Produktmodell-Zugriff“.

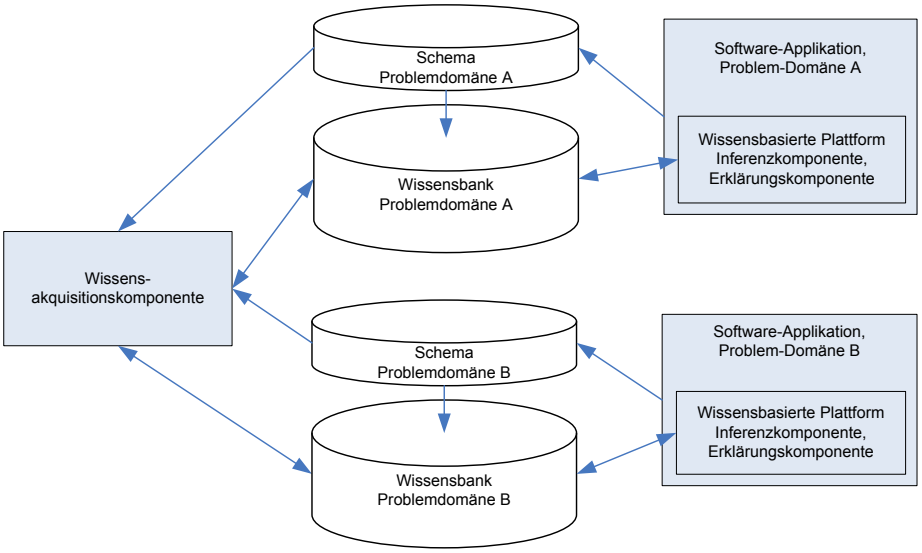


Abbildung 4-1: Wissensbasierte Plattform mit mehreren Domänen-Schemata

Die Struktur der Problemdomäne folgt hierbei den Prinzipien der Objektorientierung. Elemente, die innerhalb der Problemdomäne von einer abstrakten Basisklasse abgeleitet wurden, können in gleicher Weise auch in das Produktmodell-Schema übernommen werden.

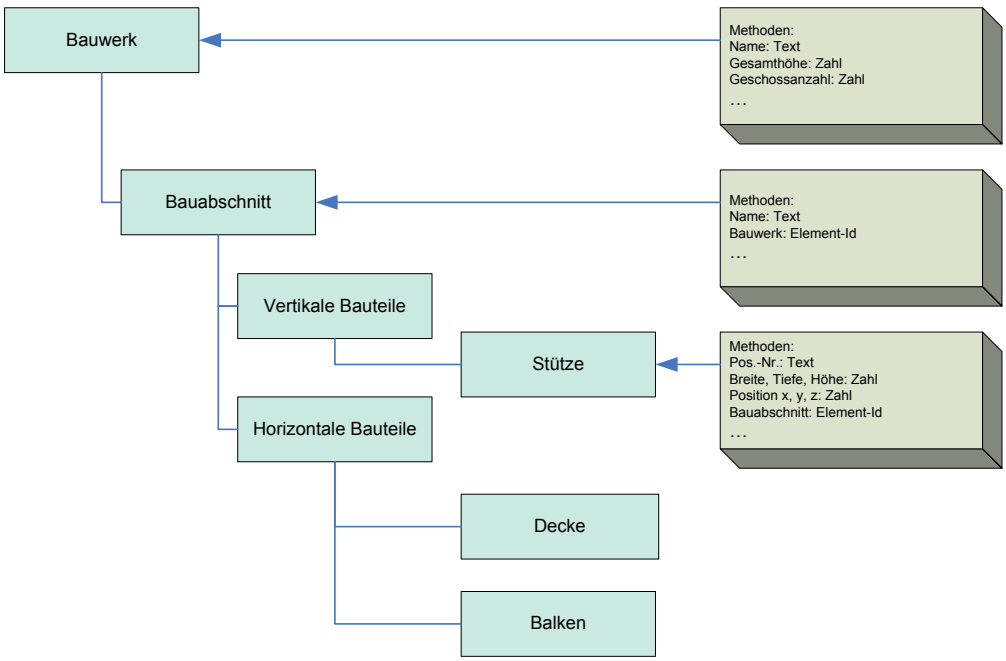


Abbildung 4-2: Beispielhaftes Domänen-Schema

Hierdurch wird ermöglicht, das Wissen in der Wissensbank übersichtlich und problemgerecht zu strukturieren und gleiche Informationen nicht mehrfach definieren zu müssen. Dies kann beispielsweise durch Anordnung einer gemeinsamen Formel innerhalb des Elementes „horizontale Bauteile“ erfolgen, das für Decken und Balken gleichermaßen anwendbar ist (Abbildung 4-2).

Gleichzeitig besteht ein Zugriff der untergeordneten Elemente auf alle ihre Vorgänger, auch wenn auf Seiten der Problemdomäne keine Vererbung im objektorientierten Sinne vorliegt. Somit ist es möglich, aus untergeordneten Bauteilen, wie beispielsweise Stützen und Decken, ohne Umwege Informationen des Bauabschnitts oder des gesamten Bauwerks abzufragen. Die Vererbung entsteht hier lediglich durch Definition der entsprechenden Berechnungselemente in der Wissensbank.

Ein grundlegendes Prinzip, dass dieser Ansatz verfolgt, ist das objektorientierte Konzept des Überladens. Dieses ermöglicht die Definition allgemeiner Berechnungselemente innerhalb übergeordneter Elemente. Für einzelne untergeordnete Elelemte, die zur Berechnung eines allgemeingültigen Parameters einen von der übergeordneten Form abweichenden Berechnungsweg erfordern, wird das fallweise „Überladen“ des zugehörigen Berechnungselementes ermöglicht.

4.1.2 Deklarative Wissensmodellierung

Die Modellierung des Wissens innerhalb einer Wissensbank erfolgt in deklarativer Form. Die Elemente einer deklarativen Wissensbasis befinden sich in loser Kopplung zueinander, wodurch keine direkten Beziehungen zwischen Wissenselementen bestehen. Die Verknüpfungen der Elemente erfolgen anhand von Meta-Informationen der Wissenselemente, wie Bezeichnern und Datentypen, und werden erst bei Bedarf durch die Inferenzmaschine zugeordnet und angewendet. Es erfolgt keine explizite Definition der Reihenfolge von Handlungsabläufen, wie dies bei prozeduraler Wissensmodellierung der Fall wäre. Die Anwendung einer Wissensbank auf eine konkrete Problemstellung erfolgt nicht wie zu erwarten mit dem „Anstoßen“ eines Berechnungsvorganges, sondern durch den Beginn der Suche nach dem gewünschten Ergebnis, was auch als „Zielgetriebene Inferenz“ bezeichnet wird. Diese Suche beginnt an definierten Elementen, den Selektionen (siehe 4.1.4.1).

Eine Selektion beinhaltet ein oder mehrere Fragestellungen, die im Fall der Anwendung zu einer konkreten Aktion oder einer definierten Rückmeldung führen. Zur Lösung dieser Fragestellungen sind meist weitere Informationen erforderlich, zu deren Lösung passende Elemente innerhalb der Wissensbank vorhanden sind. Da diese Elemente wiederum weitere Informationen benötigen, müssen auch diese gesucht und angewendet werden, wodurch sich selbständig ein komplexer Berechnungsablauf ergeben kann, der in keiner Form vorab prozedural definiert wurde.

Die deklarative Vorgehensweise hat ihre größten Stärken in der flexiblen Wissensmodellierung. Berechnungselemente können sehr leicht ausgetauscht und ihre Anwendung durch Regeln gesteuert werden, wodurch die Wartbarkeit der Wissensbanken vereinfacht wird und für den Wissensingenieur handhabbar bleibt. Die Art der Wissensrepräsentation wird in hohem Maß von den in der Inferenzkomponente zur Anwendung kommenden Schlussfolgerungsmechanismen beeinflusst.

Im Rahmen der vorliegenden Arbeit erfolgt die Inferenz in Zusammenhang mit einem vollständig automatisierten Optimierungsprozess. Dies hat zur Folge, dass während des Inferenzprozesses keinerlei Benutzerinteraktionen erforderlich sind. Der Inferenzprozess muss somit autark ablaufen und eindeutig interpretierbare Ergebnisse

produzieren. Folglich beschränken sich die Schlussfolgerungsmechanismen auf die Anwendung des monotonen Schließens, und somit der Deduktion [BIB93].

4.1.3 Element-übergreifender Informationsaustausch

Die Definition von Berechnungselementen erfolgt immer im Kontext eines Elementes des Produktmodells einer Problemdomäne. Häufig ist zudem der Zugriff auf Werte anderer Elemente erforderlich, die dem eigenen Element nicht direkt übergeordnet sind. Hierin besteht eine große Problematik bei Anwendung wissensbasierter Systeme auf dreidimensionale Planungsmodelle. Die Beschreibung geometrischer Zusammenhänge, die beispielsweise in Normen und Regelwerken relativ trivial klingt, stellt für die formale Wissensrepräsentation eine große Herausforderung dar. Inzwischen existieren mehrere geometrische Abfragesprachen [BOR07]. Die Leistungsfähigkeit dieser Sprachen für unkritische Schlussfolgerungen nimmt stetig zu, erreicht jedoch bisher nicht die erforderliche Präzision, die zur Durchführung von statisch relevanten Bemessungen erforderlich ist. Aus diesem Grund werden im Rahmen der wissensbasierten Auswertung ausschließlich konsistent modellierte Beziehungen zwischen Elementen verwendet.

Der Zugriff auf Elemente erfolgt mittels einer geteilten Variablennotation. Während beispielsweise innerhalb einer Stütze auf die eigene Breite direkt über [b] zugegriffen werden kann, wird die Breite der rechten Nachbarstütze über [StRechts.b] abgefragt. Hierzu ist es erforderlich, dass zusätzlich zur Berechnung des Wertes [b] auch für [StRechts] in der Wissensbank eine oder mehrere entsprechende Methoden vorhanden sind. Die Ergebnisse dieser Methoden beinhalten keinen numerischen Wert, sondern eine direkte Referenz auf ein fremdes Bauteil über dessen ID. In den meisten Fällen sind diese Methoden vom Typ „Produktmodell-Zugriff“, da teilweise umfangreiche Abfragen notwendig sind, die jedoch unabhängig von fachspezifischem Wissen sind und somit keiner Anpassung durch den Wissensingenieur bedürfen.

4.1.4 Berechnungselemente

Die Formalisierung des in einer Wissensbank vorhandenen Wissens muss in einer für die Wissensdomäne sinnvollen Struktur erfolgen. Die ausschließliche Abbildung von Bemessungsregeln, Nachweisen und Designvorschriften durch die Regeln der zweiwertigen Logik ist nicht praktikabel. Das von Albert in [ALB02] vorgeschlagene und im Rahmen der vorliegenden Arbeit erweiterte *Modell der Elemente einer Wissensbasis (MEW)* verwendet zur Wissensformalisierung unterschiedliche Berechnungselemente.

Eine Wissensbank gehört einer oder mehreren Problemdomänen mit der dazugehörigen Schema-Definition an. Jedes Berechnungselement der Wissensbank wird einem Element dieser Schemadefinition eindeutig zugewiesen. Somit kann sichergestellt werden, dass ein Berechnungselement nur in einem korrekten Kontext angewendet wird. Berechnungselemente sind in der Lage, unter den gegebenen Bedingungen Aussagen über einen oder mehrere Parameter zu treffen, meist unter Hinzunahme weiterer Parameter. Ausnahmen bilden lediglich die Elemente *Regelmenge* (Kap. 4.1.4.8), *Nachweis-Selektion* (Kap. 4.1.4.1) und *Design-Selektion* (Kap. 4.1.4.2). Diese dienen ausschließlich der Steuerung des Inferenzprozesses.

4.1.4.1 Nachweis-Selektion

Eine Nachweis-Selektion bildet den Startpunkt für Nachweise und damit verbundene Berechnungen, wie diese im Ingenieurwesen üblich sind. Die Nachweis-Selektion beinhaltet im Wesentlichen zwei Bedingungen: Eine Erforderlichkeits-Bedingung und

eine Erfülltheits-Bedingung. Erstere beinhaltet eine logische Regel, die überprüft, ob ein Nachweis zu führen ist. Ist dies zutreffend, wird die Erfülltheits-Bedingung angewandt, die den eigentlichen Inferenzprozess anstößt. Das Ergebnis der Erfülltheits-Bedingung signalisiert die Erfülltheit oder Nicht-Erfülltheit des zugehörigen Nachweises.

4.1.4.2 Design-Selektion

Die Design-Selektion ist eine wesentliche Neuerung des MEW, die im Rahmen der vorliegenden Arbeit entwickelt wurde. Die Aufgabe der Design-Selektion besteht in der wissensbasierten Erstellung einzelner Instanzen der Problemdomäne. Im Beispiel der Tragwerksplanung führt die Anwendung einer Design-Selektion beispielsweise zur Erstellung von Bauteilen wie Stützen und Deckenelementen. Die Anordnung in Bezug auf das Schema der Problemdomäne erfolgt innerhalb von Elementen eines übergeordneten Modells, beispielsweise eines Topologiemodells. Das Ergebnis einer vollständig ausgewerteten Design-Selektion kann somit als detaillierte Handlungsanweisung zur Erstellung einzelner Bauteile innerhalb des Topologiemodells angesehen werden.

Eine Design-Selektion benötigt eine Reihe von Informationen, die zur Erstellung eines Bauteils notwendig sind. Die Entscheidung, ob eine Design-Selektion an einer bestimmten Stelle eines Topologiemodells anwendbar ist, wird durch Auswertung einer Anwendungsbedingung getroffen. Weitere erforderliche Parameter, wie beispielsweise die Lage und Dimension des Bauteils, werden im Rahmen des Inferenzprozesses ermittelt.

4.1.4.3 Konstante

Konstanten sind keine Berechnungselemente im eigentlichen Sinne, dienen jedoch der Wissensstrukturierung und besseren Wartbarkeit der Wissensbank. Konstanten können neben numerischen Werten auch Texte oder boolesche Aussagen (Ja/Nein) repräsentieren. Wie bei allen Berechnungselementen kann auch die Anwendung einer Konstanten durch die Erfülltheit einer Regel bedingt sein.

4.1.4.4 Formel

Mathematische Formeln und Ausdrücke werden mit Hilfe des Berechnungselementes „Formel“ formalisiert. Die für den Anwender übliche Art der Formeldefinition erfolgt in Textform. Zu deren Interpretation wurde ein eigener Formelparser entwickelt. Der Formelparser ermöglicht die Eingabe von Konstanten, Operatoren, definierten mathematischen Funktionen sowie von Variablen.

Da der Vorgang der Textinterpretation einen nicht unerheblichen Rechenaufwand erfordert, wurde zur Datenablage eine objektorientierte Baumstruktur gewählt. Hierdurch muss eine eingegebene Formel nur einmalig beim Speichern durch den Parser interpretiert werden. Die spätere Anwendung durch die Inferenzkomponente operiert auf der wesentlich schneller zu verarbeitenden Baumstruktur. In Versuchsreihen konnte bei durchschnittlicher Komplexität der Ausdrücke eine Beschleunigung des Berechnungsvorganges um den Faktor 18-20 ermittelt werden.

Abbildung 4-3 zeigt den Formel-Assistenten der Wissensakquisitionskomponente „KBDT“. Am oberen Rand erfolgt die Eingabe der Formel in Textform. Darunter befindet sich eine Auflistung mathematischer Funktionen, sowie im rechten Teil eine Liste der im aktuellen Kontext verfügbaren Variablen. Der untere Textbereich ermöglicht die Verifikation der eingegebenen Formel und zeigt deren Darstellung als Objektstruktur.

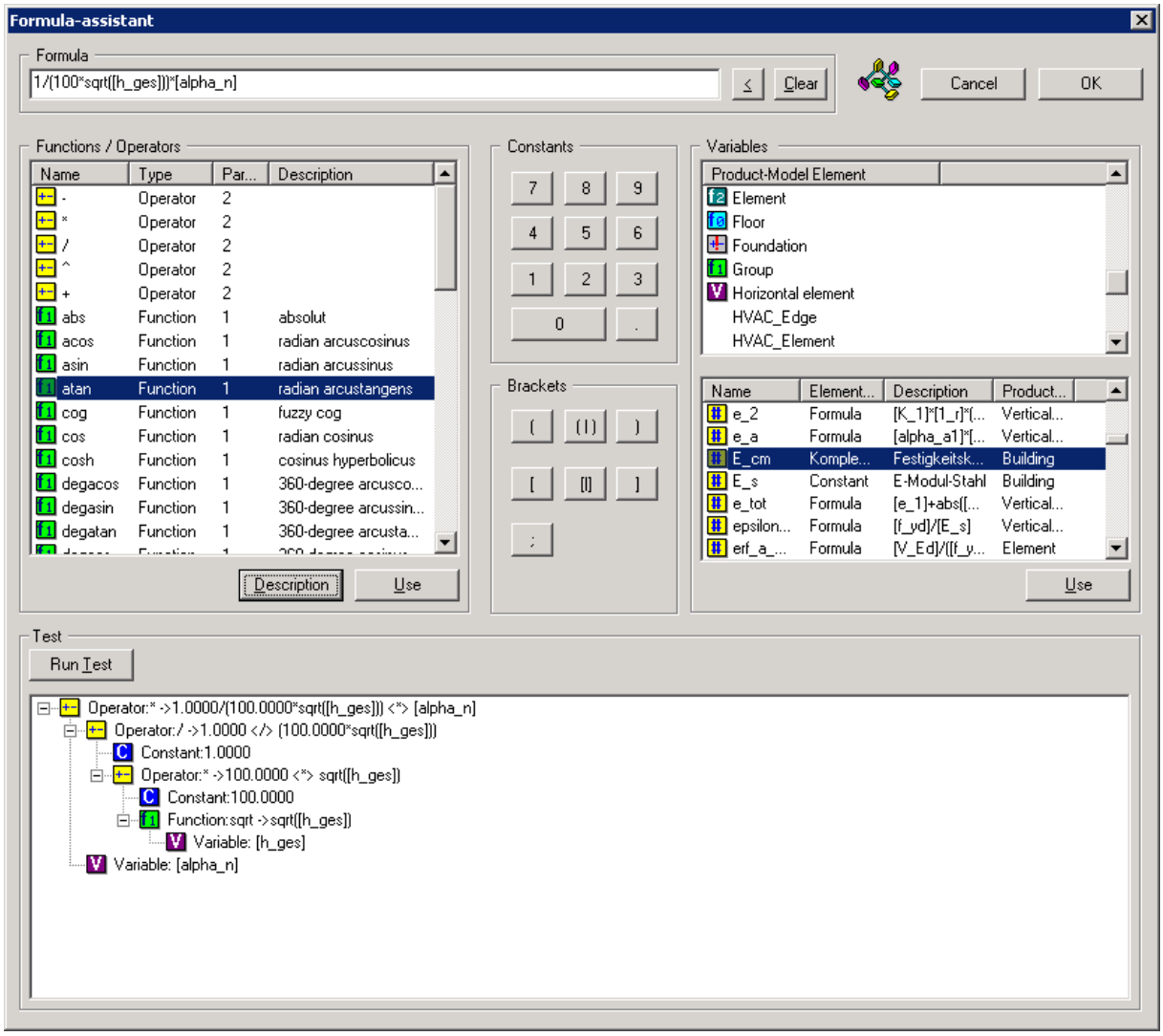


Abbildung 4-3: Formel-Assistent im „KBDT“

4.1.4.5 Diagramm

Das Berechnungselement „Diagramm“ dient der direkten grafischen Repräsentation von Diagrammen in einer Form, die auch in technischen Regelwerken üblich ist. Die in Diagrammen enthaltenen Werte könnten gleichwohl numerisch formalisiert werden, was jedoch sowohl den Prozess der Wissensakquisition als auch die spätere Nachvollziehbarkeit der Ergebnisse mindert.

Die Bereitstellung einer grafischen Benutzerschnittstelle zur Eingabe von Diagrammen (Abbildung 4-4) ermöglicht dem Wissensingenieur, direkt auf einer eingescannten grafischen Vorlage zu operieren, ohne mühevoll selbst Werte aus Normblättern zu entnehmen. Die hierzu erforderliche Skalierung des Wertebereiches und Verschiebung des Ursprunges kann direkt innerhalb der Wissensakquisitionskomponente vorgenommen werden.

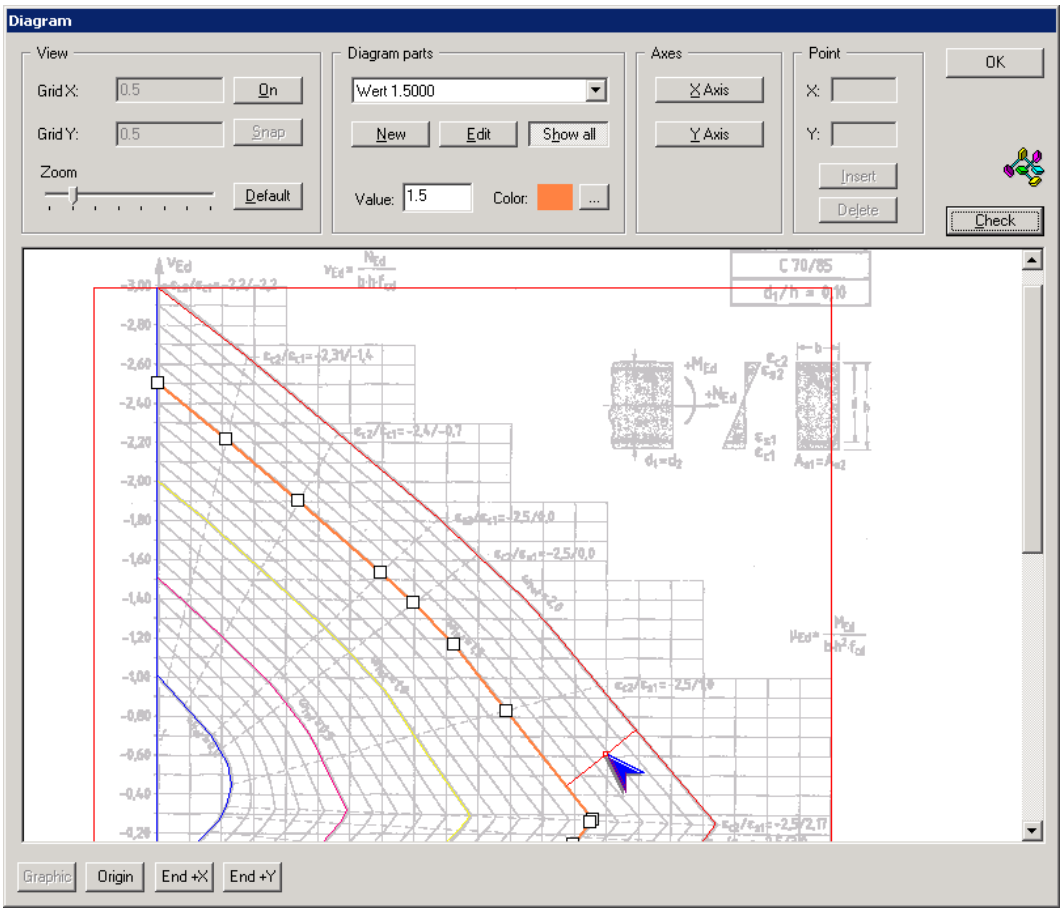


Abbildung 4-4: Vorlagenbasierte grafische Definition eines Diagramms im „KBDT“

4.1.4.6 Tabelle

Tabellen bieten eine strukturierte Sammlung von Konstanten, die anhand von ein oder zwei Eingangsgrößen selektiert bzw. interpoliert werden. Die Eingangsgrößen einer Tabelle können sowohl numerisch als auch textbasiert sein.

Komplex Table

Table axles: Axle X (results) | Axle Y (input values) | Check | OK

Betonfestigkeitsklasse / ->	f_ck	f_ck_cube	f_cm	f_ctm	f_ctk_0_05	f_ctk_0_95	
C12/15	12.0000	15.0000	20.0000	1.6000	1.1000	2.0000	2580
C16/20	16.0000	20.0000	24.0000	1.9000	1.3000	2.5000	2740
C20/25	20.0000	25.0000	28.0000	2.2000	1.5000	2.9000	2880
C25/30	25.0000	30.0000	33.0000	2.6000	1.8000	3.3000	3050
C30/37	30.0000	37.0000	38.0000	2.9000	2.0000	3.8000	3190
C35/45	35.0000	45.0000	43.0000	3.2000	2.2000	4.2000	3330
C40/50	40.0000	50.0000	48.0000	3.5000	2.5000	4.6000	3450
C45/55	45.0000	55.0000	53.0000	3.8000	2.7000	4.9000	3570
C50/60	50.0000	60.0000	58.0000	4.1000	2.9000	5.3000	3680
C55/67	55.0000	67.0000	63.0000	4.2000	3.0000	5.5000	3780
C60/75	60.0000	75.0000	68.0000	4.4000	3.1000	5.7000	3880
C70/85	70.0000	85.0000	78.0000	4.6000	3.2000	6.0000	4060

Abbildung 4-5: Definition einer Tabelle im „KBDT“

Tabellen können auch mehr als einen Ausgangswert liefern. In diesem Fall darf jedoch nur die vertikale Achse als Eingangsgröße definiert werden. Ein Beispiel zur Anwendung von Tabellen ist die Definition von Betonfestigkeitswerten (Abbildung 4-5).

4.1.4.7 Produktmodell-Zugriff

Der Produktmodell-Zugriff bildet die Schnittstelle zwischen den statischen Attributen des Produktmodells und dem Inferenzmechanismus. In der Schema-Definition der Problemdomäne werden für jedes Schema-Element dessen bereitgestellte öffentlichen Methoden deklariert, die durch Produktmodell-Zugriffe erreichbar sind. Die Methoden können neben dem einfachen Zugriff topologischer und geometrischer Informationen auch komplexere Zugriffsfunktionen bereitstellen. Diese sollten jedoch allgemeingültig und unabhängig von Domänen-spezifischem Wissen sein, welches in der Wissensbank zu formalisieren ist.

Durch Parametrisierung können Produktmodellzugriffe zudem flexibel angewendet werden. Beispielsweise erfolgt die Berechnung der Gesamtkosten eines Bauwerkes unter Anderem durch Summenbildung der Einzelkosten der Bauteile. Eine entsprechende öffentliche Methode „Summe“ innerhalb des Bauwerks erlaubt die namentliche Angabe eines Parameters, der über alle Bauteile aufsummiert und mittels eines Produktmodellzugriffs innerhalb des Bauwerkes bereitgestellt wird.

4.1.4.8 Regelmenge

Eine Regelmenge beschreibt eine Sammlung von abhängigen Anwendungsbedingungen. Jede der Prämissen verweist in der zugehörigen Konklusion auf ein oder mehrere Berechnungselemente, die nur bei Erfülltheit der jeweiligen Prämisse zur Anwendung kommen. Bei Erfülltheit einer Prämisse gelten automatisch alle weiteren Prämissen als nicht erfüllt. Sollte keine der Prämissen erfüllt sein, gelten die Berechnungselemente einer Default-Konklusion als anwendbar.

Jede der Prämissen einer Regelmenge ist abhängig von der Auswertung weiterer Berechnungselemente. Daher erfolgt im Rahmen der Inferenz lediglich die Auswertung der Prämissen, die erforderlich sind, um die Anwendbarkeit des abhängigen Berechnungselementes zu bestimmen.

4.1.5 Wissensakquisitionskomponente

Die Definition und Anpassung der Wissensbanken erfolgt mittels einer eigenständigen Wissensakquisitionskomponente, dem *Knowledge Base Definition Tool (KBDT*), dargestellt in Abbildung 4-6. Auf der linken Seite sind zwei Baumstrukturen zu erkennen, die der Strukturierung der Berechnungselemente nach Typ und Domänen-Schema dienen. Der obere Baum dient der Selektion der Berechnungselemente nach ihrer Art. Der untere Baum repräsentiert das Elementschema der Problemdomäne und wurde unabhängig von der Implementierung der wissensbasierten Plattform ausschließlich durch Import der Schema-Definition (siehe Kap. 4.1.1) generiert. Auf der rechten Seite werden Berechnungselemente angezeigt, die jeweils die Schnittmenge der gewählten Selektion in den Baustrukturen erfüllen.

4.1.6 Inferenzkomponente

Die Inferenzkomponente dient der Anwendung des in einer Wissensbank formalisierten Wissens auf konkrete Problemstellungen. Als Problemstellung wird hier beispielsweise ein aus einzelnen Bauteilen bestehendes Tragwerksmodell angesehen. Auf einer höheren Abstraktionsebene entspricht das Tragwerksmodell einem Prob-

lemraum, der sich aus einer hierarchischen Komposition einzelner Elemente zusammensetzt. Das oberste Element wäre das Tragwerk. Möglicherweise existieren untergeordnete Elemente, wie Bauabschnitte oder Stockwerke. Die untere Ebene bilden die einzelnen Bauteile, wie Stützen und Wände. Die Attribute des Tragwerksmodells stellen strukturiertes Faktenwissen des Problemraumes dar, das dem Inferenzprozess zugrunde liegt.

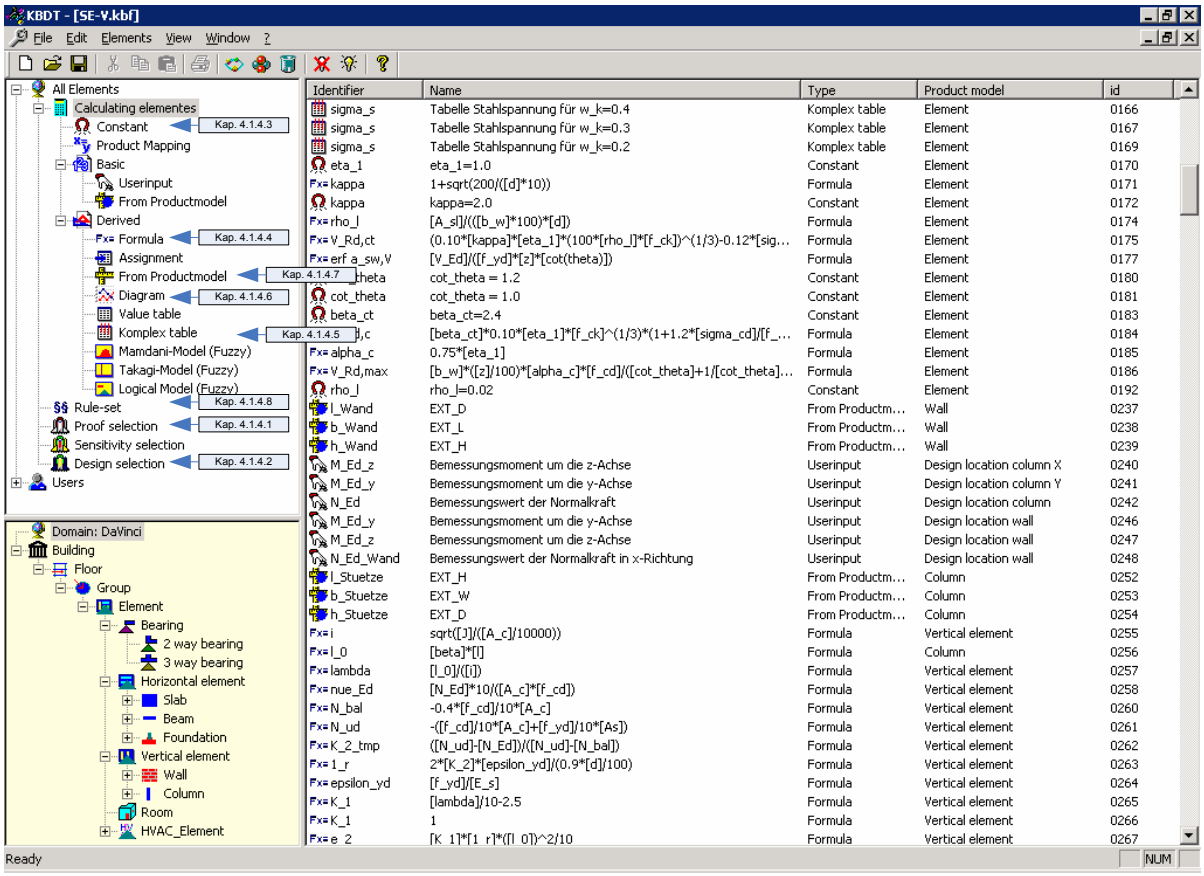


Abbildung 4-6: Wissensakquisitionskomponente „KBDT“

Bedingt durch die deklarative Wissensrepräsentation erfolgt ein zielgetriebener Inferenzprozess. Die Inferenz wird durch Anwendung einer oder mehrerer Selektionen ausgelöst. Die in den Selektionen beinhalteten Fragestellungen nach Erfordernis oder Erfülltheit sind abhängig von Parametern, die sich mit Hilfe weiterer formalisierter Berechnungselemente ermitteln lassen. Die Anwendung der Selektionen erfolgt nicht global, sondern getrennt für jedes Element des Problemraumes. Ebenso enthält jedes Element eine eigene Liste mit Fakten, die im Rahmen der Inferenz erweitert wird.

Bei der Inferenzkomponente handelt es sich um ein Agenda-basiertes System. Grundlage dieser Systeme ist eine dynamische Agenda im Sinne einer To-Do-Liste, die sämtliche im weiteren Verlauf des Inferenzprozesses zu ermittelnden Informationen enthält. Die Agenda wird während des Inferenzprozesses schrittweise „gefüllt“, bis einzelne Elemente durch Anwendung bekannten Faktenwissens ermittelbar sind. Die rekursive Abarbeitung der Agenda führt letzten Endes zur Ermittlung der durch eine auslösende Selektion geforderten Informationen.

4.1.6.1 Kontrollstrategie

Die Kontrollstrategie bestimmt, welche Berechnungselemente in den jeweiligen Fällen zur Anwendung kommen. Jedes Berechnungselement unterliegt hierbei per Definition folgenden Einschränkungen:

1. **Lösungskompetenz:** Berechnungselemente enthalten Meta-Informationen zur Klassifikation der Informationen, die durch diese Elemente ermittelbar sind, anhand eindeutiger Bezeichner und Datentypen.
2. **Schema-Bindung:** Berechnungselemente sind nur innerhalb bestimmter Typen von Elementen des Problemraumes gültig.
3. **Einschränkung durch Regeln:** Zusätzliche Anwendungsregeln in Form von Regelmengen können die Anwendung von Berechnungselementen beschränken

Die klassischen in Frage kommenden Kontrollstrategien sind „Breitensuche (engl.: breadth first search)“ und „Tiefensuche (engl.: depth first search)“. Die Breitensuche verfolgt zunächst das Ziel, auf einer Ebene möglichst viele Lösungswege mit geringer Tiefe zu finden. Bei Mehrdeutigkeit wird derjenige Lösungsweg mit der geringsten Tiefenstruktur gewählt. Die Tiefensuche verfolgt das konträre Ziel und untersucht jeden möglichen Lösungspfad bis in die unterste Ebene. Der erste derart als vollständig lösbar klassifizierte Pfad wird bei der Tiefensuche als Lösungsweg gewählt.

Der in der vorliegenden Arbeit entwickelte wissensbasierte Ansatz weicht von klassischen „Expertensystemen“ derart ab, dass zum Zeitpunkt der Inferenz keinerlei Benutzerinteraktion erwünscht ist, gleichzeitig jedoch eindeutig interpretierbare Ergebnisse generiert werden. Der Forderung nach Eindeutigkeit folgend muss bereits auf der ersten zu untersuchenden Ebene eine eindeutige Lösungsmöglichkeit ermittelbar sein, da andernfalls nicht sichergestellt wäre, dass bei Untersuchung weiterer Pfade immer genau ein gültiger Pfad ermittelt würde. Als Suchalgorithmus kommt daher ausschließlich die Breitensuche zum Einsatz. Bereits bei der Definition der Wissensbank muss sichergestellt werden, dass für jede Problemstellung eine eindeutige Lösungsmöglichkeit formalisiert wurde.

4.1.6.2 Abhängigkeits- und Einflussnetz

Während des Inferenzprozesses erfolgt die Berechnung von Parametern, die jeweils einzelnen Elementen des Problemraumes zugeordnet sind. Deren Ermittlung basiert größtenteils auf weiteren Parametern derselben oder anderer Elemente. Die relationale Beziehung ist für den reinen Inferenzprozess nach abgeschlossener Ermittlung eines jeden Parameters unerheblich. Zur transparenten Nachvollziehbarkeit des Inferenzprozesses und zur minimalen Neuberechnung bei Änderungen ist die Modellierung der Abhängigkeiten jedoch zwingend erforderlich.

Jeder Parameter enthält daher Relationen zu den Parametern, die zu seiner Ermittlung herangezogen wurden, und von deren Wert er somit abhängig ist. Ebenso sind für jeden Parameter dessen Einflüsse auf andere Parameter relational modelliert. Somit ergibt sich ein umfassendes Abhängigkeits- und Einflussnetz, das zur Analyse der Auswirkungen nachträglicher Veränderungen oder im Rahmen der Erklärungskomponente Verwendung finden kann.

4.1.7 Erklärungskomponente

Die Erklärungskomponente ist ein wesentlicher Bestandteil der wissensbasierten Plattform. Mit ihrer Hilfe erlangt der Anwender detaillierte Informationen über die berechneten Parameter in jedem Element des Problemraums.

Die Erklärungskomponente stellt für jedes Element des Problemraumes entsprechende Informationsseiten zur Verfügung, die detaillierte Informationen über Parameter und ausgeführte Entwurfs- und Nachweisselektionen enthalten. Die Darstellung der Erklärungskomponente erfolgt über eine eingebettete Version eines Internet-Browsers. Durch Hypertext-Verweise bei den einzelnen Parametern sind jeweils detaillierte Informationen verfügbar. Die Erzeugung der HTML-Seiten erfolgt vollständig dynamisch mit dem Zeitpunkt der jeweiligen Anforderung. Hierdurch wird unnötiger Ressourcenverbrauch vermieden, der bei Vorberechnung sämtlicher möglicher Informationsseiten in Abhängigkeit von der Anzahl der Elemente und der Parameter des Problemraums erheblich sein könnte.

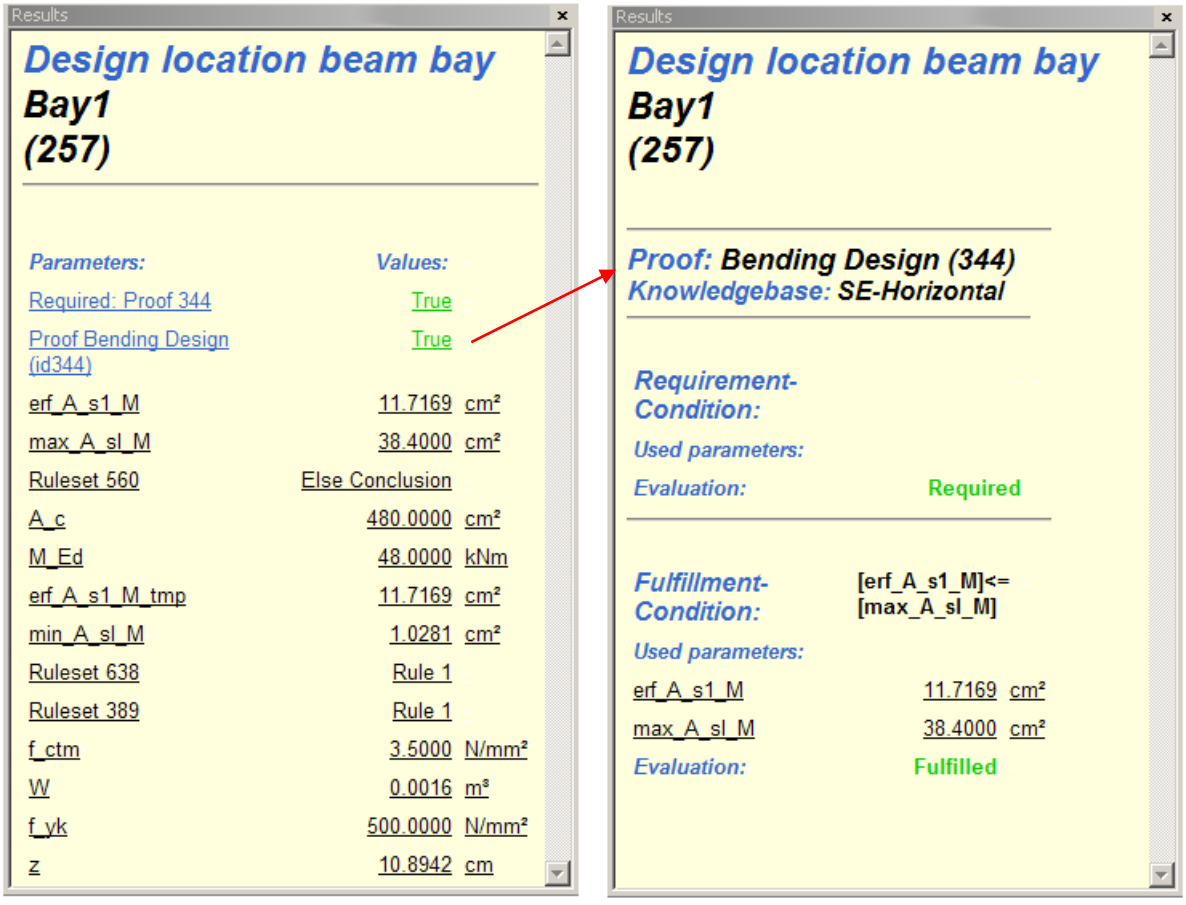


Abbildung 4-7: Erklärungskomponente der wissensbasierten Plattform

Die in der Erklärungskomponente dargestellten Informationen beinhalten sowohl allgemeine, in der Wissensbank definierte Parameter, als auch konkret während des Inferenzprozesses ermittelte Werte. Ebenso werden Abhängigkeiten von anderen Parametern sowie Beeinflussungen weiterer Parameter anhand des Abhängigkeits- und Einflussnetztes dargestellt. Durch direkte Verlinkung der Abhängigkeiten wird für den Anwender eine schnelle, lückenlose Navigation von einem Berechnungsergebnis bis hin zu sämtlichen Eingangsgrößen ermöglicht.

Die Elemente des Problemraumes, die weitgehend aus geometrischen Elementen bestehen, enthalten mindestens folgende Informationen:

- Bezeichnung
- Typ (Stütze / Decke / Stockwerk / Gruppe etc.)
- Positionsnummer
- Geometrie (Abmessungen, Lage, sofern vorhanden)
- Liste von Parametern, die während des Inferenzprozesses ermittelt wurden

Zu jedem Parameter sind folgende Informationen verfügbar (vgl. Abbildung 4-7):

- Wissensbank, die zur Erzeugung des Parameters geführt hat
- verwendetes Berechnungselement samt detaillierter Beschreibung aus der Wissensbank
- aktueller Wert / Inhalt / Einheit
- Status (Gültigkeit)
- Liste der Parameter, die von diesem Parameter abhängig sind (generiert aus dem Abhängigkeitsnetz)
- Liste der Parameter, die zur Berechnung dieses Parameters verwendet wurden (generiert aus dem Einflussnetz)

4.1.8 Software-Design

In Abbildung 4-8 ist die Klassenstruktur der wissensbasierten Plattform dargestellt. Die darin enthaltenen Komponenten bilden das Berechnungselementemodell. Sie sind die Schnittmenge, auf der sowohl die Wissenserwerbskomponente als auch die Inferenzkomponente aufbauen. Die Schnittstellen zu Elementen des Problemraumes werden ausschließlich durch Verweis auf Elemente der Schema-Definition des Problemraumes hergestellt und durch die Inferenzkomponente ausgewertet. Die Klassen des Berechnungselementemodells sind vollkommen frei von Bindungen an eine Problemdomäne.

In Abbildung 4-9 ist die Klassenstruktur mit den wesentlichen Klassen der Inferenzkomponente dargestellt. Zunächst ist hieraus ersichtlich, dass die Inferenzmaschine in einen globalen und mehrere lokale Anteile aufgeteilt wurde. Ähnlich einem agentenbasierten Ansatz (wobei auf die Abgrenzung hier nicht näher eingegangen wird) bearbeitet jede lokale Inferenzmaschine genau ein Element des Problemraumes, das ihr konstant zugeordnet ist. Ebenso existiert zu jeder lokalen Inferenzmaschine eine eigene Agenda, die bisher ungelöste Aufgaben enthält.

Bei Zugriff einer Inferenzmaschine auf Werte anderer Elemente des Problemraumes erfolgt eine Anfrage an die zugehörige lokale Inferenzmaschine des jeweiligen Elementes. Möglicherweise wird zu diesem Zweck eine lokale Inferenzmaschine instan-

ziiert. Die lokale Inferenzmaschine des Fremdelementes behandelt die Anfrage nach einem Wert nach dem gleichen Schema wie Anfragen aufgrund einer Selektion.

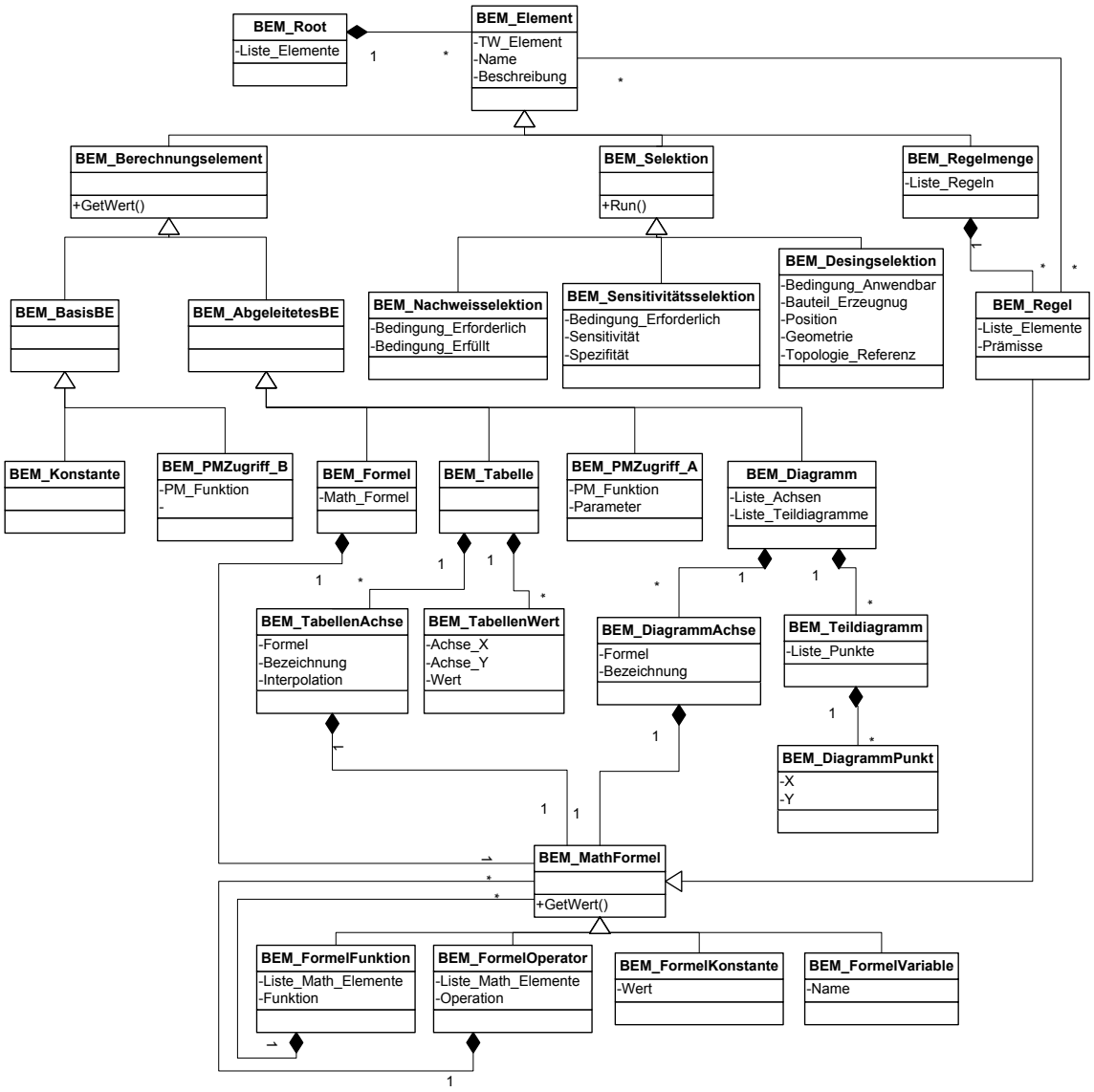


Abbildung 4-8: Klassenstruktur des Berechnungselementemodells

Der deklarative Inferenzprozess ist beispielhaft in Abbildung 4-10 dargestellt. Dieses Beispiel umfasst zur Verdeutlichung den Zugriff auf Werte mehrerer Elemente untereinander. Im vorliegenden Fall wird die Inferenz für eine Stütze durchgeführt.

Der Inferenzprozess beginnt mit der Suche nach erforderlichen Nachweisen, die durch Berechnungselemente des Typs "Nachweis-Selektion" definiert sind. In der Wissensbank ist beispielhaft ein vereinfachter Nachweis der Knicksicherheit formalisiert. Die Erfülltheitsbedingung der Nachweisselektion erfordert die Parameter b und h. Der Wert für b kann direkt aus der Stütze über den Produktmodellzugriff "dx" ermittelt werden. Zur Berechnung der Höhe h existiert in der Wissensbank für den Elementtyp "Stütze" die Formel "h=Stockwerk.h". Die Punkt-Notation besagt, dass der Parameter "h" nicht aus der betrachteten Stütze, sondern aus einem Element des

Typs "Stockwerk" verwendet wird. Hierbei ist der Verweis auf "Stockwerk" ein weiterer Parameter der Stütze.

Zur Berechnung der Formel ist es zunächst erforderlich, eine Methode zur Ermittlung des Parameters "Stockwerk" zu finden. Hierzu existiert für die Stütze ein entsprechender Produktmodellzugriff, der einen Verweis auf das der Stütze übergeordnete Stockwerk liefert. Nach Ermittlung des Verweis-Parameters für "Stockwerk" erfolgt die Ermittlung des Parameters "h" innerhalb des (aus Sicht der Stütze fremden) Elementes "Stockwerk". Nachdem sowohl für "b" als auch für "h" die erforderlichen Werte ermittelt wurden, erfolgt die Bestimmung der Erfülltheit des Nachweises durch Auswertung der Erfülltheitsbedingung, die initial als Startpunkt der Berechnung diente.

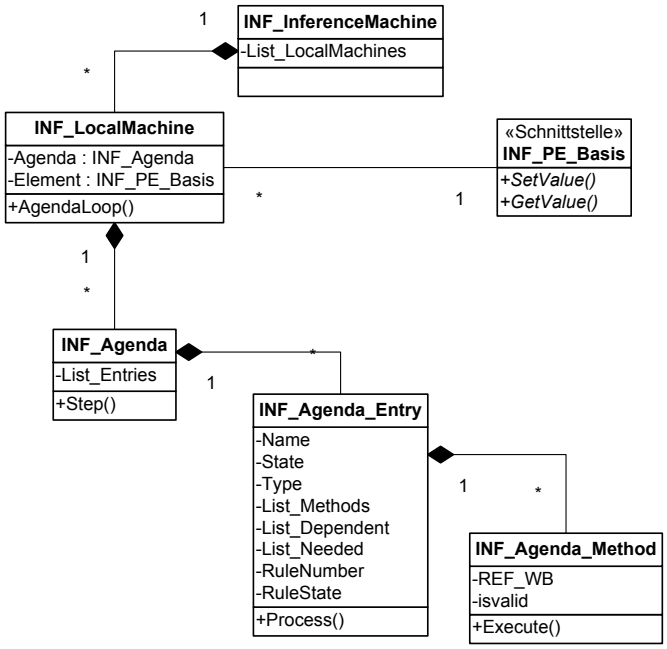


Abbildung 4-9: Klassenstruktur der Inferenzkomponente

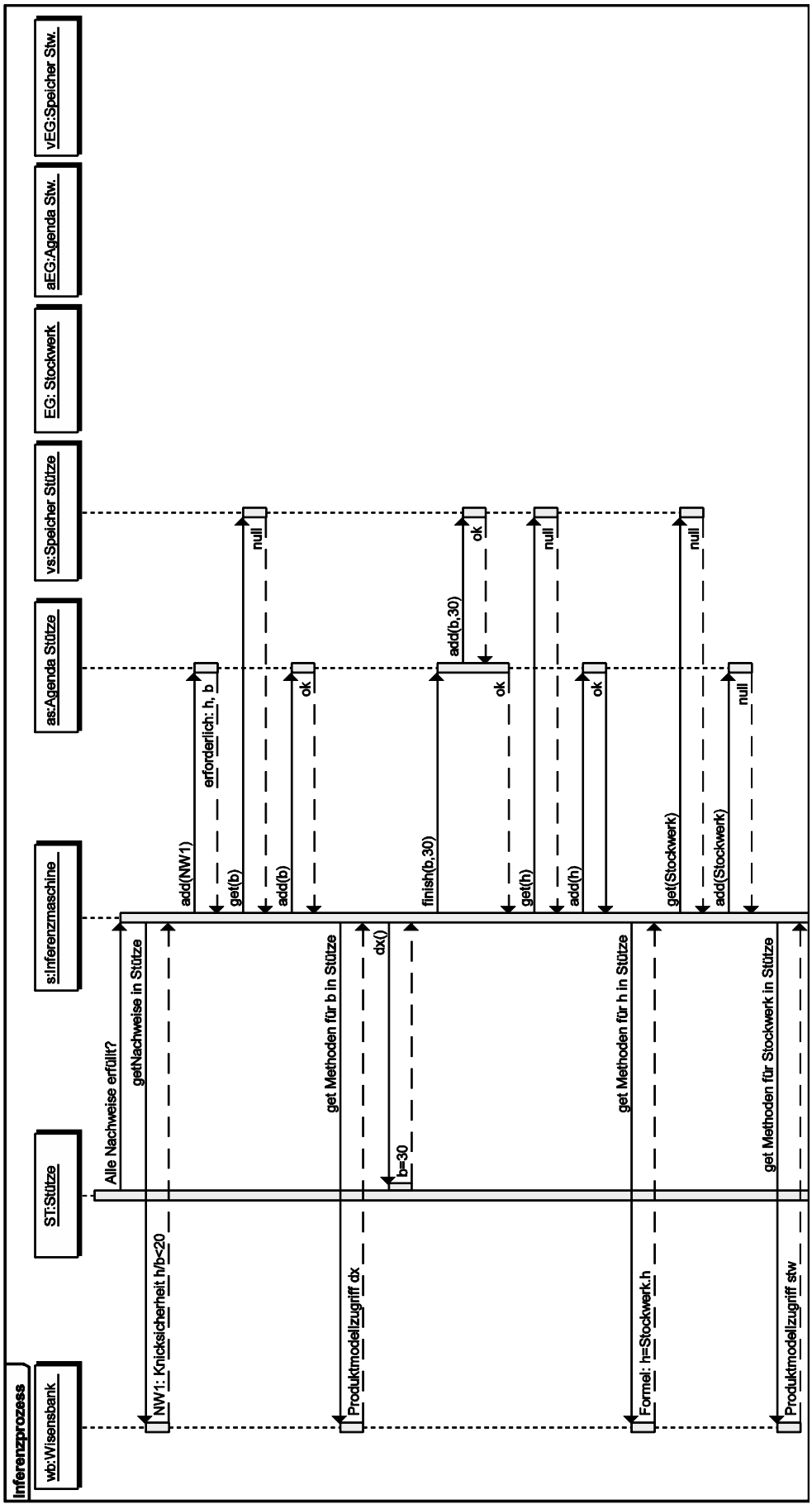


Abbildung 4-10: Inferenzprozess über zwei Strukturelemente

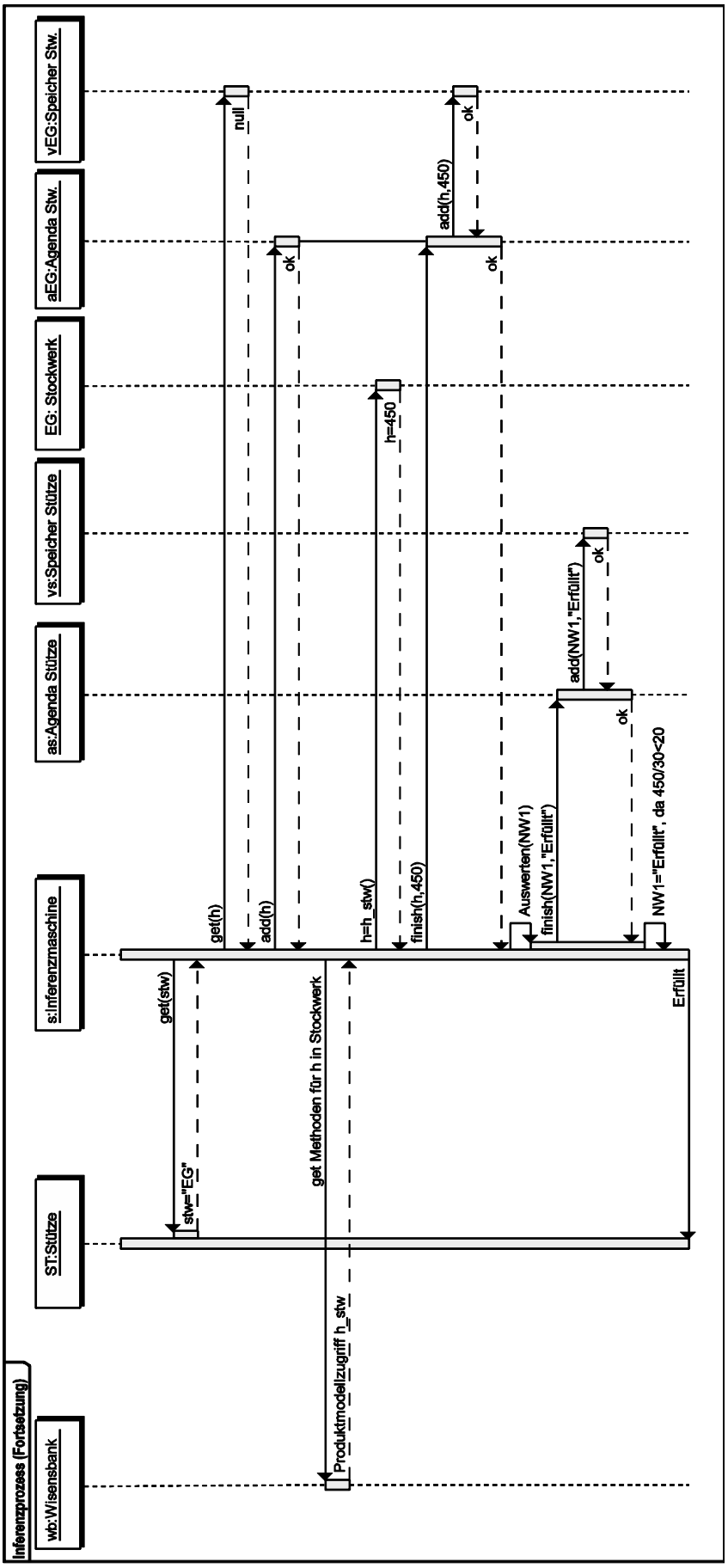


Abbildung 4-11: Inferenzprozess über zwei Strukturelemente (Fortsetzung)

4.2 Optimierungskomponente

Das im Rahmen der vorliegenden Arbeit entwickelte Optimierungsverfahren basiert auf dem Konzept der genetischen Programmierung. Genetische Programmierung wurde ursprünglich mit der Intention entwickelt, zur Lösung eines Optimierungsproblems die Individuen mittels einfacher „Programme“ in der Sprache LISP zu generieren und zu optimieren [KOZ92]. Die der Sprache LISP zugrunde liegende Grammatik war dabei wohlbekannt und das Vokabular der möglichen Ausprägungen der terminierten und nicht terminierten Knoten sowie deren Zuordnungsregeln eindeutig definiert. Die praktische Anwendung genetischer Programmierung auf Aufgaben des Ingenieurwesens ist in den meisten Fällen nicht praktikabel durch LISP-Programme zu repräsentieren. Auch eine beliebige andere starr definierte Form der Repräsentation erscheint nicht sinnvoll. Vielmehr muss sich die Form der Repräsentation an der jeweiligen Problemstellung orientieren.

Eine Möglichkeit besteht darin, bei der Entwicklung einer Komponente zur genetischen Optimierung die Grammatik der Problemstellung direkt in den Programmcode zu implementieren. Die Implementierung ist hierbei auf die konkrete Problemstellung beschränkt und nicht ohne weiteres auf andere Problemstellungen anwendbar. Die logische Abbildung der genetischen Programmierung kann relativ komplex sein. Durch eine rein problemspezifische Implementierung erhöht sich die Zahl der potentiellen Fehlerquellen deutlich. Gerade unter diesem Gesichtspunkt sollte die Wiederverwendung eines stabilen Programmcodes erfolgen. Das im Rahmen dieser Arbeit entwickelte Softwarekonzept verfolgt zudem das Ziel größtmöglicher Flexibilität. Die hierzu notwendige Fähigkeit zur Adaption beschränkt sich nicht nur auf das in Wissensbanken formalisierte Entwurfs- und Bemessungswissen, sondern soll auch die freie Gestaltung der Individuen ermöglichen. Auch aus diesem Grund ist eine flexible und kontextfreie Definition der Grammatik zur Steuerung der Optimierungskomponente erforderlich.

4.2.1 Bildungsvorschrift

Die im Rahmen dieser Arbeit entwickelte Optimierungskomponente verwendet eine modifizierte Version der *Backus-Naur Form (BNF)* zur Definition von Bildungsvorschriften. Dadurch wird die Notation beliebiger syntaktischer Elemente für unterschiedlichste Anwendungsfälle ermöglicht. Zur Reduktion der Komplexität werden im Folgenden Beispiele zur Optimierung von mathematischen Funktionen dargestellt. Die beschriebenen Verfahren lassen sich jedoch ohne weiteres auf komplexere Optimierungsprobleme übertragen, was im Rahmen der vorliegenden Arbeit erfolgte.

Die Backus-Naur Form ist eine kontextfreie Metasprache zur Definition von regulären Grammatiken. Sie entspricht dem Typ 2 der Chromsky Hierarchie [ROZ97] und wird unter Anderem zur Notation gängiger Programmiersprachen verwendet [KNU64]. Da nach [ROS96] und [MON94] die Bildungsvorschriften einer stark typisierten genetischen Programmierung weitgehend einer formalen Grammatikdefinition entsprechen, eignet sich die BNF grundsätzlich zur Notation der Regeln einer Optimierungsaufgabe.

Listing 4-1 zeigt eine einfache Definition einer mathematischen Formel in der BNF. Dieses Beispiel beginnt mit einem Startsymbol <S>. Die in spitzen Klammern eingeschlossenen Symbole bezeichnen *nicht-terminierte Symbole*, die lediglich als Platzhalter dienen und in den späteren Texten nicht enthalten sind. Nicht-terminierte Symbole, die innerhalb eines Ausdruckes verwendet werden, müssen an anderer Stelle definiert sein. Dabei ist es in der BNF durchaus üblich, rekursive Definitionen

zu verwenden, wie im folgenden Beispiel das Symbol <NN>. Die weiteren Symbole sind *terminierte Symbole*, die den eigentlichen Text darstellen. Der vertikale Trenner separiert die möglichen Alternativen voneinander.

```
<S>:=y=<RN>*<Fx>
<Fx>:=<F>(<RN>*x)
<F>:=sin|cos|tan
<RN>:=<URN>|-<URN>            //Reellwertige Zahl
<URN>:=<NN>|<NN>.<NN>         //Reellwertige Zahl,
                              //vorzeichenlos
<NN>:=<N>|<NN><N>             //Ganzzahl
<N>:=0|1|2|3|4|5|6|7|8|9      //Einzelne Ziffer
```

Listing 4-1: Definition einer mathematischen Formel in der BNF

Durch rekursive Anwendung der Bildungsvorschrift aus Listing 4-1 können unterschiedliche Ausprägungen in Form einer Baumstruktur erzeugt werden. Jede dieser Ausprägungen repräsentiert im Rahmen der genetischen Programmierung ein mögliches Individuum.

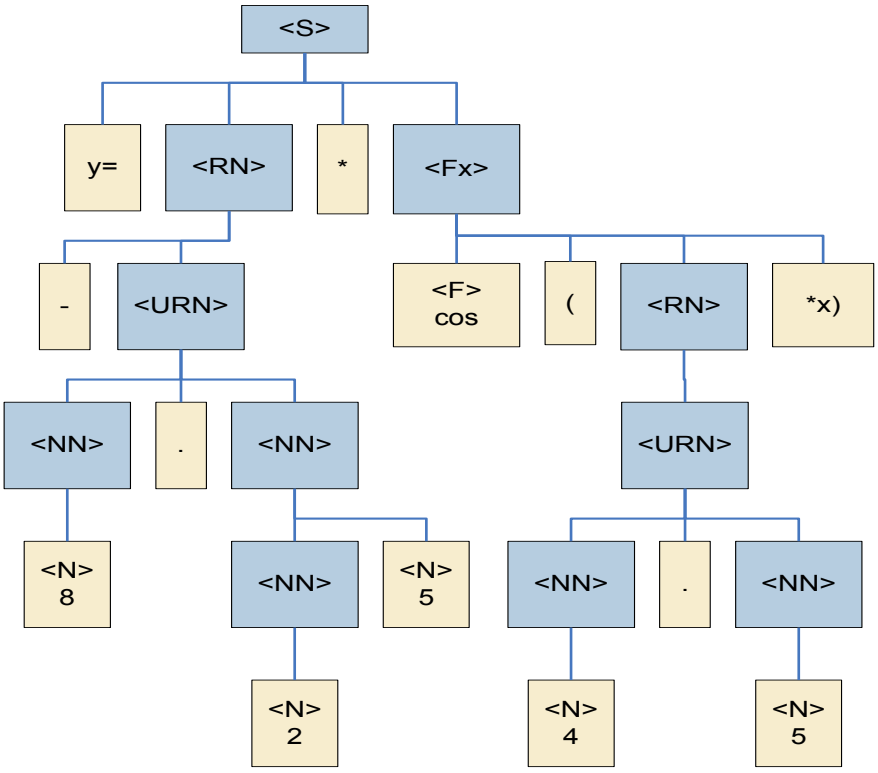


Abbildung 4-12: Mögliche Ausprägung der BNF-Definition aus Listing 4-1

Die Inhalte der Baumstruktur jedes Individuums können in ihrer Gesamtheit als ein Programmcode interpretiert werden, der in Bezug auf Grammatik und Vokabular der

zugrunde liegenden BNF entspricht. Die Übersetzung eines derart repräsentierten Individuums in einen Programmcode erfolgt durch einfache Verkettung der terminierten Symbole innerhalb des Baumes. Abbildung 4-12 zeigt eine mögliche Ausprägung der definierten Bildungsvorschrift. Listing 4-2 enthält den Programmcode zu diesem Beispiel sowie zu weiteren möglichen Individuen.

Für die Operationen der Kreuzung und Mutation ist jedoch weiterhin die Baum-Repräsentation erforderlich, da die Rücktransformation des Programmcode in die Baumstruktur nur möglich ist, wenn der Programmcode eindeutige Merkmale zur Zuordnung der zugrunde liegenden Struktur aufweist. Da dies bei komplexeren Definitionen nicht mehr möglich ist, ergeben sich Probleme bei zielgerichteter Initialisierung einer Anfangspopulation. Hier ist es erforderlich, die Individuen in Form ihrer Baum-Repräsentation zu definieren.

```
y=-8.25*cos(4.5*x)          //Beispiel aus Abbildung 4-12
y=14.225*sin(12.98*x)
y=-19.8*tan(2*x)
y=3406982035203598.1234*cos(-0.00005*x)
```

Listing 4-2: Programmcode möglicher Ausprägungen der BNF-Definition aus Listing 4-1

Diese Beispiele verdeutlichen die Möglichkeiten der BNF, zeigen aber auch wichtige Einschränkungen in Bezug auf Optimierungsprobleme des Ingenieurwesens auf. So ist beispielsweise die Notation von Zahlenwerten unzureichend. Durch die rekursive Notationsform besteht zunächst keine Möglichkeit, den Definitionsbereich sowie die Anzahl der numerischen Stellen einzuschränken. Besonders nachteilig wirkt sich dieses Verhalten bei Kreuzungs- und Mutations-Operationen aus. Hier entsteht in der Praxis oft das so genannte „code-growing“ oder „bloat“ [KOZ94], wobei die Anzahl der Knoten stetig wächst, ohne dabei eine wirkliche Verbesserung zu bewirken (vgl. Kap. 4.2.7).

4.2.1.1 Wertebereich-Symbole

Eine wesentliche Erweiterung der BNF, die im Rahmen dieser Arbeit entwickelt wurde, ist die Notation numerischer Werte, wie in Listing 4-3 gezeigt. Die übergebenen Parameter entsprechen hierbei den Unter- und Obergrenzen des Definitionsbereiches sowie der Schrittweite. Durch hinreichend kleine Wahl der Schrittweite können beliebig hohe Genauigkeiten erzeugt werden. Sehr vorteilhaft erweist sich das Verhalten von numerischen Werten bei Kreuzungs- und Mutationsoperationen, da je Zahlenwert lediglich ein einzelner Knoten erzeugt wird. Speziell angepasste Operatoren stellen zudem sicher, dass Zahlenwerte entsprechend behandelt werden.

Die Abgrenzung der Wertebereich-Symbole zwischen terminierten und nicht-terminierten Symbolen ist nicht eindeutig. Einerseits weisen diese Symbole keine untergeordneten Symbole auf, wodurch sie keine nicht-terminierten Symbole sind. Andererseits beinhalten sie auch keine starren Informationen, wie dies bei terminierten Symbolen üblich ist. Die Abgrenzung ist insofern wichtig, da die genetischen Operationen für terminierte und nicht-terminierte Symbole unterschiedlich anwendbar sind. So ist es beispielsweise wirkungslos, ein terminiertes Symbol „A“ mit einem terminierten Symbol „A“ vom selben Typ zu kreuzen, da im Ergebnis keine Änderun-

gen auftreten würden. Im Fall von Wertebereich-Symbolen können genetische Operationen sinnvoll angewendet werden, da deren Wert variabel ist.

```
<S>:=y=<RN>*<Fx>
<Fx>:=<F>(<RN>*x)
<F>:=sin|cos|tan
<RN>:=[-10;10;0.1]
```

Listing 4-3

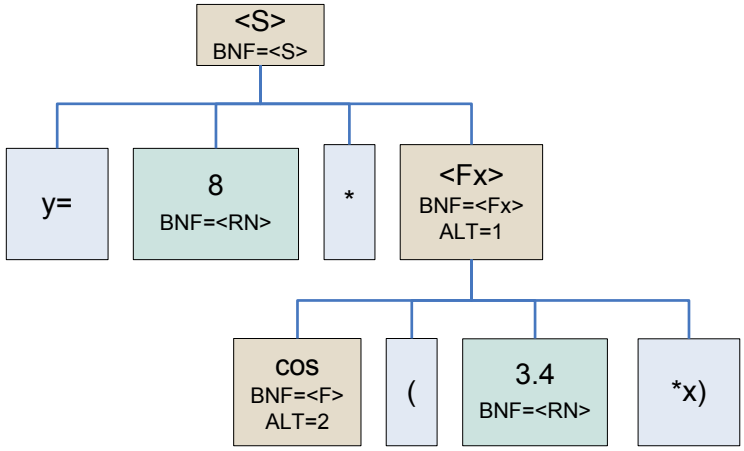


Abbildung 4-13: Repräsentation eines Individuums mit Wertebereich-Symbolen

Die basierend auf einer BNF-Definition erzeugten Individuen (Abbildung 4-12) können durch die Einführung von Wertebereich-Symbolen in ihrer Komplexität wesentlich reduziert werden (Abbildung 4-13). Die einzelnen Knoten des Baumes eines Individuums beinhalten neben der notwendigen Information zur Erstellung des Programmcodes weitere Attribute, die im Rahmen der Optimierung erforderlich sind. Diese repräsentieren Herkunftsmerkmale, die in Bezug zu der zugrunde liegende BNF-Grammatikdefinition stehen. Verschiedene genetische Operationen erfordern das Wissen über den Ursprung eines Knotens, einerseits zur Zuordnung knotenspezifischer Parameter, andererseits zur Identifikation von möglichen Kreuzungs-Kandidaten oder zur Steuerung der Mutation.

```
y=8*cos(3.4*x)
```

Listing 4-4 zu Abbildung 4-13

4.2.1.2 Symbolspezifische Optimierungsparameter

Die für den Optimierungsprozess essentiellen Operationen *Kreuzung* und *Mutation* erfordern bei Anwendung auf Individuen der genetischen Programmierung die Selektion bestimmter Knoten. Üblicherweise erfolgt eine rein zufällige Selektion, so dass jeder Knoten die gleiche Selektionswahrscheinlichkeit aufweist. Dies kann sich sehr

ungünstig auf das Optimierungsverhalten auswirken. Da durch Definition der Bildungsvorschrift mittels der BNF genauere Informationen über jeden Knotentyp definierbar sind, besteht die Möglichkeit, den Knoten entsprechend dem zu erwartenden Optimierungsverhalten abweichende Optimierungsparameter zuzuweisen.

Im Rahmen der vorliegenden Arbeit wurde die BNF um Symbolspezifische Optimierungsparameter erweitert. Diese bestehen zunächst aus Gewichtungsfaktoren für jede der relevanten genetischen Operationen Rekombination (siehe Kap. 4.2.5), reguläre Mutation (siehe Kap. 4.2.6.1) sowie probabilistische Mutation (siehe Kap. 4.2.6.2):

- Selektionsgewicht für Rekombination
  (selection weight crossover - SWC)
- Selektionsgewicht für reguläre Mutation
  (selection weight regular mutation - SWRM)
- Selektionsgewicht für probabilistische Mutation
  (selection weight probabilistic mutation - SWPM)

Die Entscheidung, ob eine genetische Operation auszuführen ist, wird ausschließlich über globale Optimierungsparameter (beispielsweise die Rekombinationsrate) gesteuert. Die Symbolspezifischen Selektionsgewichte finden erst während der tatsächlichen Anwendung der genetischen Operation Verwendung, um die zufällige Auswahl geeigneter Symbole innerhalb eines Individuums zielgerichtet zu beeinflussen. Diese im Rahmen der vorliegenden Arbeit entwickelte Erweiterung trägt wesentlich zur Effizienz der genetischen Operationen bei.

Die Parameter beinhalten neben individuellen Mutations- und Rekombinationsgewichten auch Gewichte (IW) für die Initialisierung der Alternativen der nichtterminierten Symbole:

- Initialisierungsgewicht (initialization weight - IW)

So kann beispielsweise der Funktion "sin" des Symbols <F> ein höheres Initialisierungsgewicht zugeordnet werden, falls bei Definition der BNF bekannt ist, dass die Wahrscheinlichkeit für die Sinus-Funktion als Zielfunktion höher eingeschätzt werden kann als für die beiden Alternativen.

Listing 4-5 zeigt beispielhaft die Definition einer Formel unter Berücksichtigung individueller Optimierungsparameter. Aus der Wahl der symbolspezifischen Parameter lässt sich deutlich das beabsichtigte Verhalten ableiten. Beispielsweise würde eine Rekombination am Startsymbol <S> keinen Sinn ergeben, da dies einem vollständigen Austausch der Erbinformationen der Eltern-Individuen gleich käme. Daher wurde für das Startsymbol der Wert SWC=0 gewählt. Aus dem gleichen Grund sind Mutationen an dem Startsymbol <S> nicht zielführend und infolge der gewählten Parameter SWRM=0 und SWPM=0 gänzlich ausgeschlossen. Die Wahrscheinlichkeit der Rekombination für einen Knoten vom Typ <F> ist hier genau 1,7 mal so hoch wie für einen Knoten vom Typ <RN>, während ein Knoten des Typs <Fx> nur eine Rekombinationswahrscheinlichkeit von 0,5 erhält.

Die Wahl des Parameters SWRM zur Steuerung der regulären Mutation erfolgt sehr dosiert und in Abhängigkeit davon, in welchem Abstand zum Startsymbol sich die Knoten eines Symboltyps vorrangig ausbilden werden. Je näher sich ein Knoten dem Startsymbol befindet, desto größeren Einfluss hat eine reguläre Mutation auf einen solchen Knoten, da ab diesem Knoten der gesamte folgende Teilbaum zerstört und neu initialisiert wird. In Listing 4-5 zeigt sich dies beispielsweise in der Wahl des Parameters SWRM=0,3 für das Symbol <FX>.

```
<S>:=y=<RN>*<Fx>, SWC=0.0, SWRM=0.0, SWPM=0.0
<Fx>:=<F>(<RNX>*x), SWC=0.5, SWRM=0.3, SWPM=0.0
<F>:=sin{IW=3}|cos{IW=2}|tan{IW=0.5}, SWC=1.7, SWRM=1.0,
                                      SWPM=0.0
<RN>:=[-10;10;0.1], SWC=1.0, SWRM=1.0, SWPM=1.0
<RNX>:=<RN>, SWC=1.0, SWRM=1.0, SWPM=1.5
```

Listing 4-5

Der Parameter SWPM zur Steuerung der probabilistischen Mutation wäre bei den Symbolen <S> und <FN> grundsätzlich wirkungslos, da diese nach dem Verfahren der probabilistischen Mutation nicht mutierbar sind. Gleiches gilt für das Symbol <F>, dessen drei alternativen Ausprägungen keine geordnete Reihe darstellen. Die verbleibenden Symbole <RN> und <RNX> sind inhaltlich identisch, weisen jedoch unterschiedliche Werte des Parameters SWPM auf.

Auf die Unterscheidung der Mutationsarten in „regulär“ und „probabilistisch“ wird in Kapitel 4.2.6 näher eingegangen.

4.2.2 Initialisierung

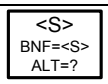
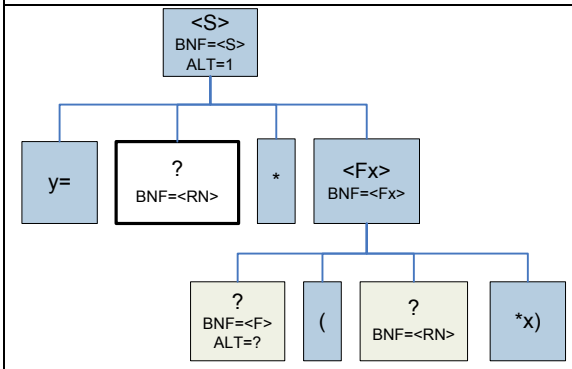
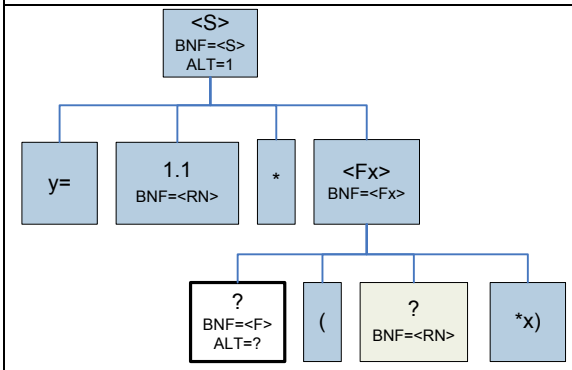
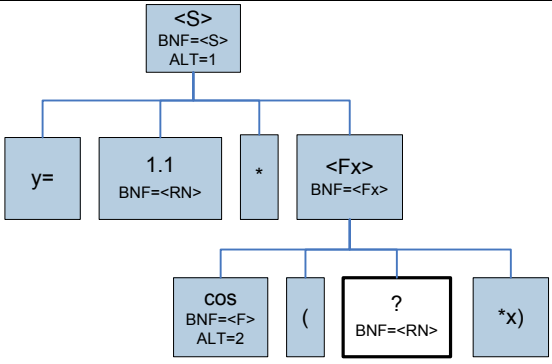
Zu Beginn einer Optimierung muss eine erste Population von Individuen in erster Generation erstellt werden. Diese *Initialpopulation* kann je nach Anwendungsfall unterschiedlichen Ursprungs sein. Möglicherweise liegen bereits gute Individuen einer vorangegangenen Optimierung vor. Dem Anwender steht auch frei, basierend auf Einschätzungen und Erfahrungen einige als gut angesehene Lösungen manuell vorzugeben. In diesen Fällen ist eine manuelle Initialisierung sinnvoll. Sofern keine Erkenntnisse über mögliche Initialindividuen vorliegen, bietet sich eine rein zufällige Initialisierung der Initialpopulation an. Oftmals liegen jedoch bereits unvollständige Informationen vor, die nicht ausreichen, um manuell zu einer sinnvollen Lösung zu gelangen. In diesen Fällen stellt eine zielgerichtete zufällige Initialisierung - die p-Initialisierung - eine geeignete Mischform zwischen manueller und zufälliger Initialisierung dar.

4.2.2.1 Manuelle Initalisierung

Der Begriff der manuellen Initialisierung impliziert einen Prozess, der alleine dem Anwender überlassen bleibt. Tatsächlich bedeutet „manuell“ in diesem Zusammenhang, dass die Individuen in beliebiger Weise außerhalb der Optimierungskomponente erzeugt und bereitgestellt werden. Ob es sich hierbei um eine manuelle Benutzerschnittstelle oder eine mit Vorwissen ausgestattete Fremdapplikation handelt, bleibt zunächst dem Anwender (dem anwendenden Entwickler) der Optimierungs-

komponente überlassen. Allerdings muss in jedem Fall verifiziert werden, dass sämtliche Individuen, die manuell initialisiert werden, der gültigen Bildungsvorschrift entsprechen. Ebenso reicht es nicht aus, nur den Programmcode eines Individuums zu erzeugen. Ausschließlich die Repräsentation der Individuen in Form einer Baumstruktur ermöglicht deren Einsatz im Rahmen der Optimierung.

Die einfachste, jedoch auch gleichzeitig fehleranfälligste Möglichkeit der manuellen Initialisierung ist die direkte Erzeugung eines Individuums als Baumstruktur. Für einen externen Ersteller bedeutet dies jedoch, dass die zugrunde liegende BNF-Grammatikdefinition exakt eingehalten werden muss, andernfalls wird ein Individuum durch eine interne Prüfung von der Optimierungskomponente abgewiesen. Dabei muss auch sichergestellt sein, dass jeder Knoten des erzeugten Individuums die zur Optimierung erforderlichen Attribute enthält.

Baumstruktur	**Programmcode**	**Entscheidung**
	y=?	Alternative=1
	y=?*?(?*x)	Wert=1.1
	y=1.1*?(?*x)	Alternative=2
	y=1.1*cos(?*x)	Wert=2.5

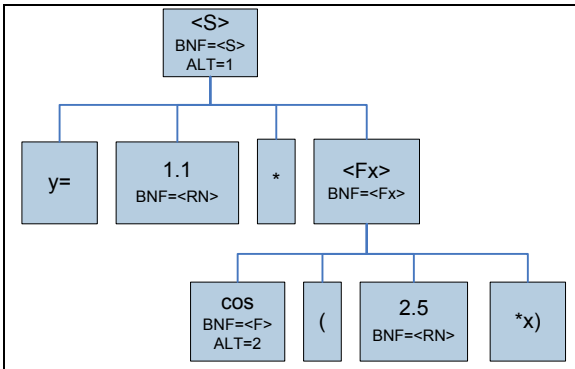
	y=1.1*cos(2.5*x)	

Tabelle 4-1: Initialisierung eines Individuums

Eine weitere Möglichkeit ist die Definition einer sequentiellen Entscheidungsliste. Diese Liste muss ebenfalls mit der Definition der BNF korrespondieren, lässt sich jedoch wesentlich einfacher durch externe Schnittstellen ansprechen. Die in der Liste vorgehaltenen Entscheidungen werden in der benötigten Reihenfolge der Verarbeitung definiert. Jede Entscheidung besteht entweder in der Auswahl einer Alternative eines nicht-terminierten Symbols, oder in der Bereitstellung eines numerischen Wertes, jeweils innerhalb der in der Bildungsvorschrift festgelegten Grenzen. Die Abarbeitung einer derartigen Entscheidungsliste erfolgt innerhalb der Optimierungskomponente. Tabelle 4-1 verdeutlicht die Auswertung einer einfachen Entscheidungsliste korrespondierend zur BNF-Definition aus Listing 4-6 (siehe Seite 65).

4.2.2.2 Zufällige Initialisierung

Im Fall der rein zufälligen Initialisierung eines Individuums erfolgt die Verarbeitung analog zur Bearbeitung einer Entscheidungsliste (Tabelle 4-1). Die Entscheidungen werden jedoch ausschließlich zufällig und gleichverteilt getroffen. Hierdurch können sämtliche möglichen Ausprägungen der BNF-Grammatikdefinition erreicht werden. Die zufällige Initialisierung erfolgt nicht nur zu Beginn einer Optimierung mit dem Zweck der Bildung einer Initialpopulation, sondern kommt auch zur Neuinitialisierung eines Teilbaumes zum Einsatz, beispielsweise durch reguläre Mutation (Kap. 4.2.6.1).

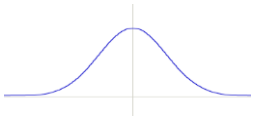
4.2.2.3 p-Initialisierung

Im Rahmen der vorliegenden Arbeit wurde eine zielgerichtete Variante der zufälligen Initialisierung entwickelt, die so genannte p-Initialisierung (p für "probabilistisch"). Diese kommt zur Anwendung, wenn für bestimmte Werte bereits Erkenntnisse vorliegen, die bezüglich ihrer Aussage nicht vollkommen gesichert oder präzisierbar sind, jedoch bereits einen guten Anhaltspunkt als Ausgangsbasis einer Optimierung bieten können.

Die Grundlage der p-Initialisierung bildet die Verwendung einer beliebigen Verteilungsfunktion, beispielsweise der Gauß'schen Normalverteilung (4-1). Der Erwartungswert μ der Standardverteilung wird hierbei dem abgeschätzten, angestrebten Wert gleichgesetzt. Die Standardabweichung σ gibt die Verteilungsvarianz an, die zur Steuerung der Streuung um den angestrebten Wert eingesetzt werden kann. Somit würde die Angabe eines Erwartungswertes von 3.0 und einer Standardabweichung von 0.5 in den meisten Fällen Werte nahe des Wertes 3.0 initialisieren, eine Stan-

dardabweichung von 4.0 hingegen resultiert in einer breiteren Streuung. Eine alternative Verteilungsfunktion ist in (4-2) gegeben.

Normalverteilung nach Gauss (nicht normiert):



$$f(x) = e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{4-1}$$

Alternative Verteilungsfunktion:

$$f(x) = \left(\left(\frac{e}{\sigma}\right)^{|x-\mu|}\right)^{-1} \quad \text{mit } \sigma < e \tag{4-2}$$

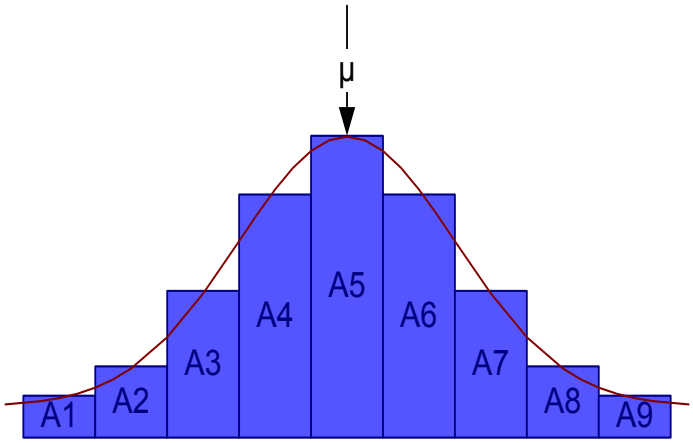


Abbildung 4-14: p-Initialisierung nicht-terminierter Symbole

Die p-Initialisierung lässt sich nicht nur auf numerische Wertesymbole anwenden, sondern ist ebenfalls auf geordnete nicht terminierte Symbole übertragbar (vgl. Kap. 4.2.6.2). In diesem Fall werden die Alternativen als ganzzahlige Werte behandelt. Hierdurch können die Initialisierungsgewichte einzelner Alternativen des nicht-terminierten Symbols entsprechend der Verteilungsfunktion belegt werden. Durch Anwendung der zufälligen Initialisierung stellt sich ein analoges probabilistisches Verhalten ein, das mit einer erhöhten Wahrscheinlichkeit die erwartete Alternative und deren Umgebung anstrebt.

4.2.3 Fitness-Bewertung

Die Bewertung der Individuen erfolgt mittels einer oder mehrerer Fitness-Funktionen, die nicht Teil der Optimierungskomponente sind, sondern als weitere Komponenten bereitgestellt werden. Im Fall der vorliegenden Arbeit besteht das Ergebnis der Fitness-Funktion aus der Gesamtheit der Bewertung eines Gebäudes. Der Bewertungsprozess durchläuft im Wesentlichen folgende Schritte:

- Interpretation des Genotypen mittels wissensbasierter Methoden
- Bildung eines dreidimensionalen Strukturmodells als Simulationsmodell
- Überführung des Strukturmodells in ein Finite Elemente Modell

- Simulation der Lasteinwirkungen aus Eigengewicht, Ausbaulast, Verkehrslast, Wind- und Erdbebenbelastung
- Ermittlung der Schnittgrößen unter den maßgebenden Lastfallkombinationen
- Wissensbasierte Bemessung und Dimensionierung der Strukturbauteile zur Verfeinerung des Simulationsmodells
- Wissensbasierte Ermittlung sämtlicher zur Bestimmung der Optimierungsparameter erforderlichen Kenngrößen
- Wissensbasierte Bestimmung der Optimierungsparameter

Die hier beschriebene Optimierungskomponente beteiligt sich am Prozess der Fitness-Evaluation lediglich durch Bereitstellung der Genotypen der Individuen als Baum oder Programmcode, Aufruf der externen Fitness-Funktion sowie Entgegennahme der Ergebnisse. Die Anzahl der Optimierungskriterien ist hierbei beliebig, so dass multikriterielle Optimierung möglich ist, auch wenn inzwischen interessante Ansätze zur Findung einer einzelnen, „idealen“ Lösung aus einer Reihe pareto-optimaler Lösungen existieren [GRI07].

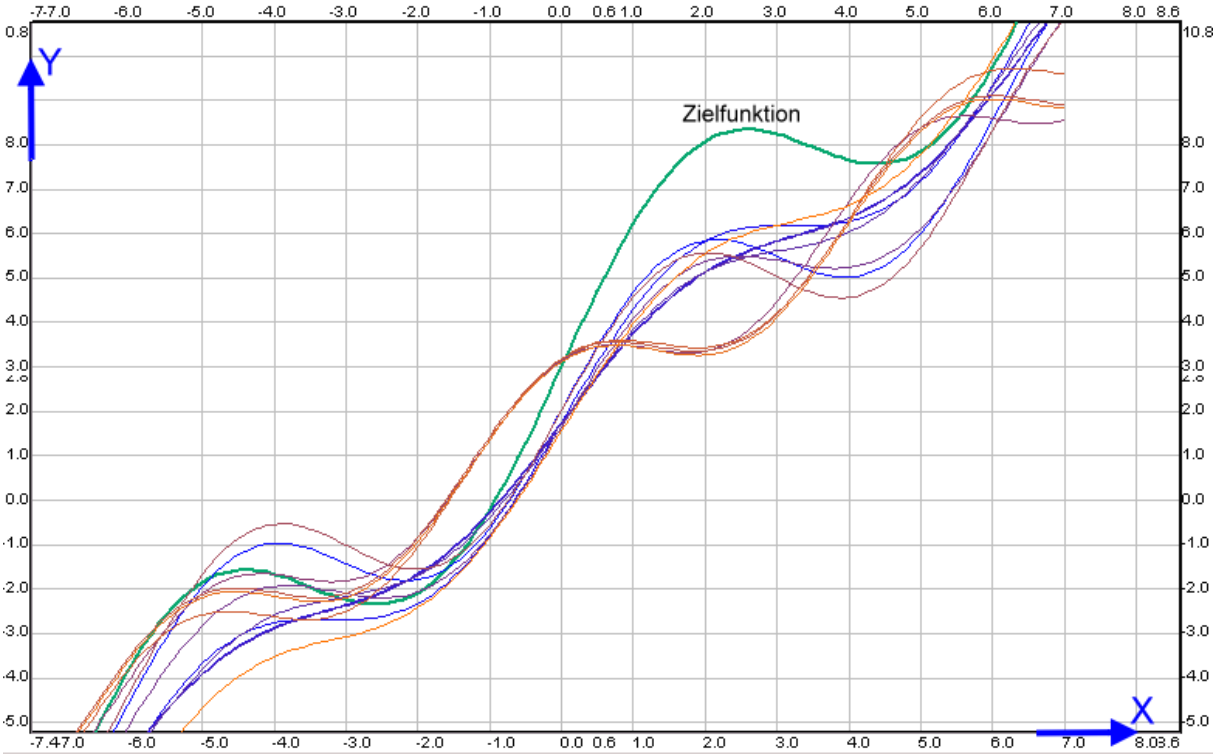


Abbildung 4-15: Zielfunktion und die bisher zwölf besten Individuen (Generation 70)

Die Fitness-Funktion des Beispiels aus Listing 4-5 ist relativ einfach bereitzustellen, weshalb dies auch für erste Experimente mit der beschriebenen Optimierungskomponente erfolgte. Zur Bewertung der Fitness werden eine gegebene Zielfunktion sowie ein zu betrachtender Definitionsbereich zugrunde gelegt. Die Fitness berechnet sich aus dem Integral der Differenz aus der Zielfunktion und der Funktion des jeweils betrachteten Individuums (4-3). Da das Individuum in der Form seines Programmcodes bereits eine mathematische Funktion darstellt, kann diese, genau wie

die Zielfunktion, direkt durch einen einfachen Parser interpretiert und berechnet werden.

Beispielhafte Fitness-Ermittlung:

$$Fitness = \int_{x1}^{x2} F_{Ziel}(x) - F_{Individuum}(x)\ dx \qquad \textbf{(4-3)}$$

Abbildung 4-15 zeigt beispielhaft die Optimierung einer umfangreichen mathematischen Funktion. Das Diagramm zeigt die Zielfunktion sowie die zwölf besten Individuen in der betrachteten 70. Generation.

```
<S>:=y=<RN>*<Fx>|y=<RN>*<Fx>+<RN>|y=<RN>*<Fx>+<RN>*x+<RN>
        SWRM=0.2, SWPM=0.0
<Fx>:=<F>(<RN>*x), SWC=0.0, SWRM=0.5, SWPM=0.0
<F>:=sin{IW=3}|cos{IW=2}|tan{IW=0.5}
<RN>:=[-5;5;0.01], SWC=2.0, SWRM=0.4, SWPM=4.0
```

Listing 4-6: BNF Grammatik-Definition zu Abbildung 4-15

```
y=2.30*sin(0.90*x)+1.42*x+3.00
```

Listing 4-7: Zielfunktion zu Abbildung 4-15

4.2.4 Selektion

Die Selektion der Individuen erfolgt basierend auf deren Fitness. Zur Sicherstellung des Erhalts der besten Individuen kommt zunächst ein Elitismus zum Einsatz. Durch Anwendung des Elitismus werden die *n* besten Individuen ohne Änderung in die folgende Generation übernommen. Die Anzahl *n* ist hierbei ein globaler Optimierungsparameter.

Zur Selektion weiterer Individuen stellt die Optimierungskomponente die drei gängigsten Verfahren zur Auswahl: *Roletterad-Selektion*, *Stochastic Universal Sampling* sowie *Turnier-Selektion* (vgl. Kap. 2.1.2.2). Für das Stochastic Universal Sampling ist die gleichzeitige Selektion aller erforderlichen Individuen notwendig. Daher werden zunächst die in erforderlicher Anzahl selektierten Individuen in einem Zwischenspeicher vorgehalten, der linear durch Kreuzungs- und Mutationsoperationen „abgearbeitet“ wird, bevor die so gewonnen Individuen der folgenden Generation zugeführt werden.

4.2.5 Rekombination

Die Rekombination der genetischen Information zweier Individuen folgt prinzipiell dem allgemeinen Verfahren der genetischen Programmierung [KOZ92]. Hierbei werden aus zwei Individuen, den Eltern, geeignete Knoten zur Kreuzung identifiziert, woraufhin die daran hängenden Unterbäume vollständig ausgetauscht werden. Die eigentliche Schwierigkeit dieses Verfahrens liegt in der Identifikation geeigneter Kno-

ten. „Geeignet“ bedeutet in diesem Zusammenhang, dass für einen Knoten des ersten Elternteils an der entsprechenden Stelle des zweiten Elternteils ein Knoten existiert (vgl. „Aligning“, [LAN02]). Im Falle von Grammatik-basierten GP Schemata, wie dem hier beschriebenen, muss zusätzlich die zugrunde liegende Bildungsvorschrift eingehalten werden, weshalb sich in diesem Fall ein anderes Vorgehen empfiehlt.

Die Bildungsvorschrift definiert für jeden Knotentyp, welche Ausprägungen dieser annehmen kann sowie welche weiteren Knoten er möglicherweise beinhaltet. Die BNF enthält keine Möglichkeit, einen Knotentyp zu definieren, der abhängig von seiner Lokalisierung innerhalb des Individuums verschiedene Ausprägungen annehmen kann. Unterschiedliche Knoten, die einer gemeinsamen BNF-Regel folgen, müssen daher auch beliebig gegeneinander austauschbar sein.

Diesem Umstand folgend kann jeder Knoten aus einem Individuum A, der nach BNF-Regel <F> aufgebaut wurde, mit jedem Knoten in Individuum B, der ebenfalls der Regel <F> entspringt, ausgetauscht werden, ohne diese Regel zu verletzen. Somit ist garantiert, dass regelkonforme Individuen nach einer Rekombination weiterhin regelkonform sind. Potentiell ist jeder Knoten eines Individuums ein „Kandidat“ für eine Rekombination, sofern von seinem Typ in beiden Elternteilen jeweils mindestens ein Knoten existiert. Zwei Knoten sind genau dann von gleichem „Typ“, wenn sie auf derselben Produktionsregel innerhalb der BNF basieren.

Wie bereits in Kap. 4.2.1.2 erläutert, wurde im Rahmen der vorliegenden Arbeit eine signifikante Erweiterung der klassischen BNF vorgenommen, die zur effizienteren Anwendung von BNF-basierten Bildungsvorschriften im Rahmen genetischer Programmierung beiträgt. Hierbei enthält die Bildungsvorschrift für jede BNF-Regel Symbolspezifische Selektionsgewichte. Durch diese Gewichte kann definiert werden, mit welcher Wahrscheinlichkeit Knoten eines bestimmten Typs zur Rekombination selektiert werden. Unabhängig vom Selektionsverfahren der Optimierung (siehe Kap. 4.2.4) erfolgt im Rahmen der vorliegenden Arbeit die Selektion eines Knotens aus dem ersten Elternteil nach dem Rouletterad-Verfahren. Der Anteil auf dem Rouletterad, den jeder Knoten einnimmt, entspricht dem Symbolspezifischen Selektionsgewicht, das in der Produktionsregel des Knotens festgelegt wurde. Somit erhält ein Knoten mit dem Gewicht 0 keinen Anteil auf dem Rouletterad und kann folglich nicht für die Rekombination selektiert werden.

Aus dem zweiten Elternteil wird ebenfalls ein Knoten für die Kreuzung selektiert. Die Menge der selektierbaren Knoten beschränkt sich hier auf die Knoten vom gleichen Typ des zuvor selektierten Knotens. Da in diesem Fall nur eine gemeinsame Produktionsregel vorliegt, sind die Gewichte der Knoten identisch, weshalb eine zufällige Selektion erfolgt. Wurde für jeden Elternteil ein gültiger Knoten selektiert, können diese zusammen mit ihren anhängenden Substrukturen ausgetauscht werden, wodurch zwei neue Kind-Individuen entstehen. Der zusammenfassende Ablauf der Kreuzungsoperation ist in Abbildung 4-16 ersichtlich.

Beispielhaft sind in Abbildung 4-17 zwei Elternteile als Ausgangsbasis einer Kreuzungsoperation dargestellt. In Abbildung 4-18 wurde der einzige Knoten des Symbols <Fx> ausgewählt und an diesem Knoten die Kreuzung durchgeführt. In Abbildung 4-19 erfolgte zunächst die Selektion eines Knotens vom Typ <RN> in Elternteil A. In Elternteil B sind zwei Knoten dieses Typs vorhanden, wobei zufällig der zweite Knoten gewählt wurde.

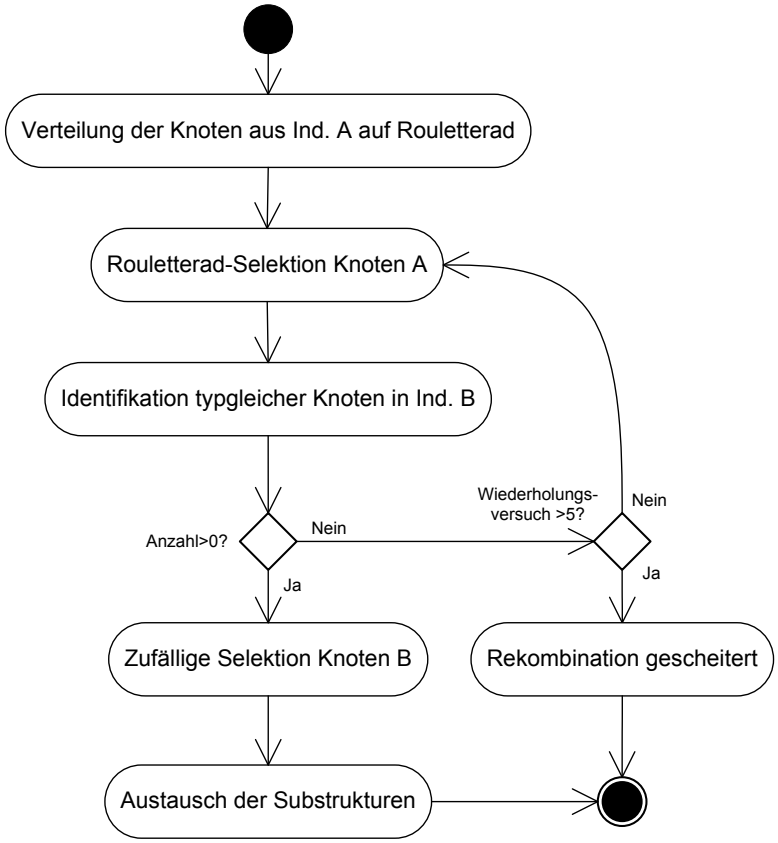


Abbildung 4-16: Ablaufdiagramm der Kreuzungs-Operation

Auch wenn das Ergebnis dieser Operation grammatikalisch korrekt ist, sind derartige Operationen nicht immer zielführend. Daher müssen mögliche Konsequenzen dieses Verhaltens bei Definition der BNF-Grammatik überprüft werden. Evtl. wäre in dem gezeigten Beispiel die Einführung einer zweiten Zahlenvariablen sinnvoll, um das zuletzt dargestellte Kreuzungsverhalten zu umgehen.

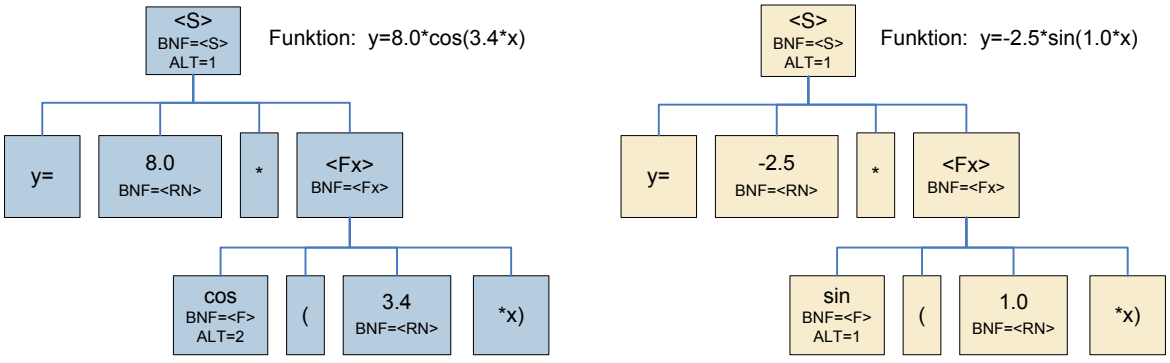


Abbildung 4-17: Elternteile A und B, Ausgangspunkt einer Kreuzung

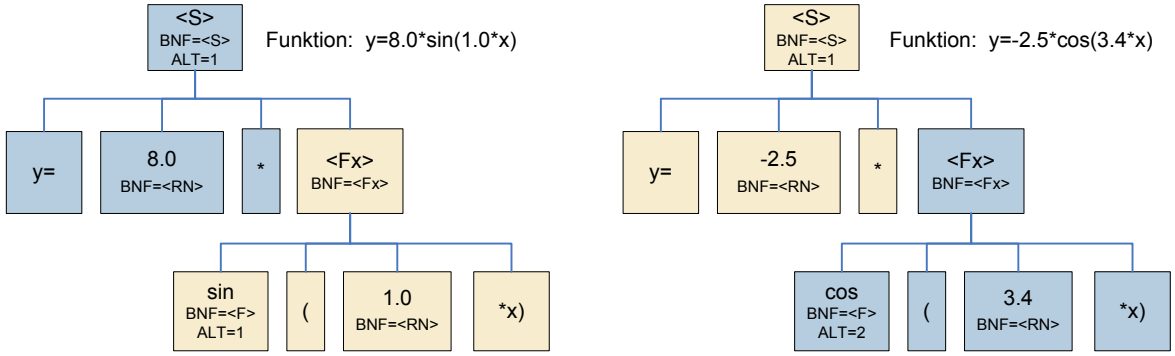


Abbildung 4-18: Kreuzung an Symbol <FX>

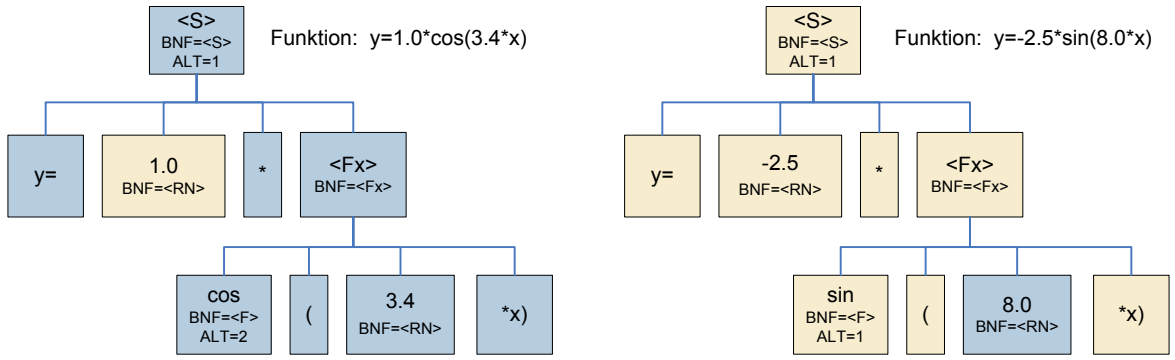


Abbildung 4-19: Kreuzung an einem Symbol <RN>

4.2.6 Mutation

Nach [POH99] sind Mutationen geringfügige, zufällige Änderungen an der genetischen Repräsentation eines Individuums. In der genetischen Programmierung ([KOZ92], [KOZ94], [LAN02]) kommt üblicherweise eine wesentlich einflussreichere Form der Mutation zur Anwendung. Teilweise erfolgen dabei relativ umfangreiche Änderungen an Individuen. Verschiedene Mutationsformen unterschiedlicher Einflussgrade können je nach Anwendungsfall zielführend sein. Gleichzeitig sind im Rahmen der entwickelten Optimierungskomponente einige Besonderheiten zu berücksichtigen, die zur Anwendung zweier unabhängiger Mutationsverfahren, „regulärer“ und „probabilistischer“ Mutation, führen.

Beiden Mutationsverfahren gemein ist die Mutationsrate, die basierend auf [POH99] sowie [BÄC93] im Normalfall mit 1/n angenommen wird, wobei *n* die Anzahl der Variablen bezeichnet. Daraus folgend wird je Mutationsoperation genau eine Variable, oder im vorliegenden Fall ein Knoten, mutiert. Diese Annahme ist dahingehend schlüssig, dass bei gleichzeitiger Veränderung mehrerer Variablen eine Verbesserung der Fitness nicht mehr einer einzelnen Änderung zuzuordnen wäre.

4.2.6.1 Reguläre Mutation

Die für genetische Programmierung übliche Form der Mutation, im Folgenden als "reguläre Mutation" bezeichnet, bewirkt Änderungen in der Baumstruktur eines Individuums durch gezielten Austausch von Knoten an zufälligen Stellen innerhalb des Baumes. Sofern es sich dabei um nicht-terminierte Knoten handelt, bedeutet die Mutation den Austausch aller Knoten, die dem betreffenden Knoten untergeordnet

sind, also eines vollständigen Teilbaumes. Dieser Teilbaum wird durch einen zufällig initialisierten Teilbaum ersetzt.

Die Auswirkungen einer regulären Mutation können sehr unterschiedlich sein, je nachdem, in welcher Tiefe des Baumes sich ein zur Mutation selektierter Knoten befindet. Aus diesem Grund wird diese Form der Mutation nur in geringem Umfang eingesetzt, um die Diversität der Individuen einer Population zu erhalten. Durch die im Rahmen dieser Arbeit eingeführte Erweiterung der BNF lässt sich die Tiefe der selektierten Knoten durch Umverteilung der knotenspezifischen Mutationsgewichte positiv steuern, so dass der Austausch zu großer Teilbäume vermieden wird.

Das folgende Beispiel soll mögliche Auswirkungen der Mutation verdeutlichen. Abbildung 4-20 stellt ein Individuum als Ausgangsbasis für eine Mutation dar. Abbildung 4-21 (links) zeigt die Mutation eines einzelnen Knotens, der mit einem neuen Wert initialisiert wurde.

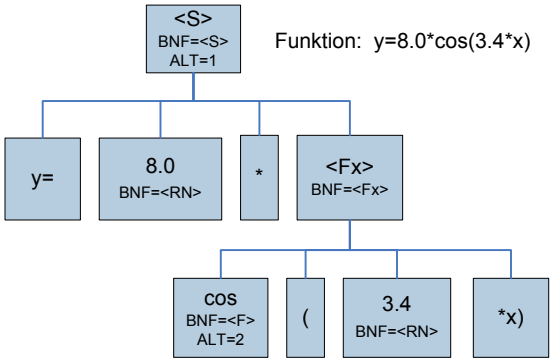


Abbildung 4-20: Selektiertes Individuum als Ausgangsbasis einer Mutation

In Abbildung 4-21 (rechts) wurde ein nicht-terminierter Knoten als Startpunkt der Mutation gewählt. Der neu initialisierte Teilbaum entspricht weiterhin der Vorgabe aus der BNF-Definition, die Neuinitialisierung bewirkt jedoch veränderte Werte in den darunter folgenden Knoten. Die maximale Auswirkung einer regulären Mutation ist in Abbildung 4-22 ersichtlich. Hier wurde als Startpunkt das Startsymbol <S> gewählt, wodurch das Individuum vollständig reinitialisiert wurde. Derartiges Verhalten ließe sich durch Definition des lokalen Mutationsgewichtes SWRM=0 für das Startsymbol vermeiden.

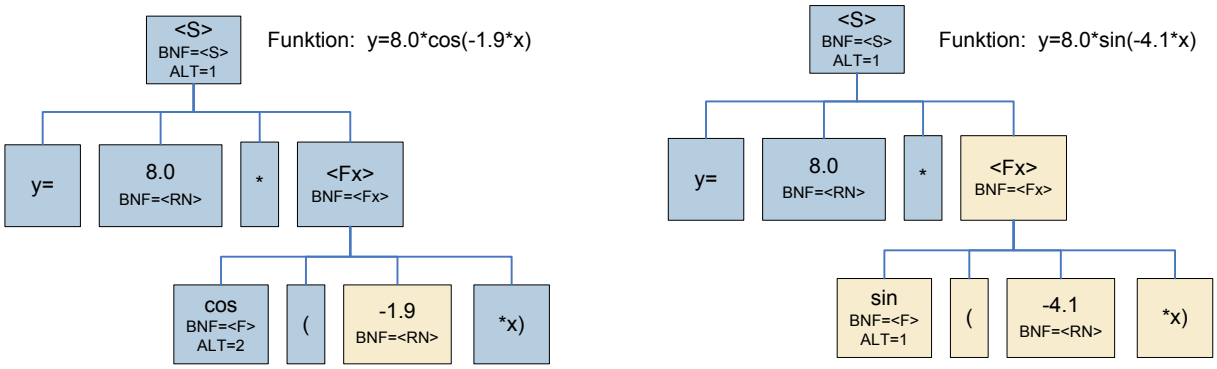


Abbildung 4-21: Zwei mögliche „Mutanten“

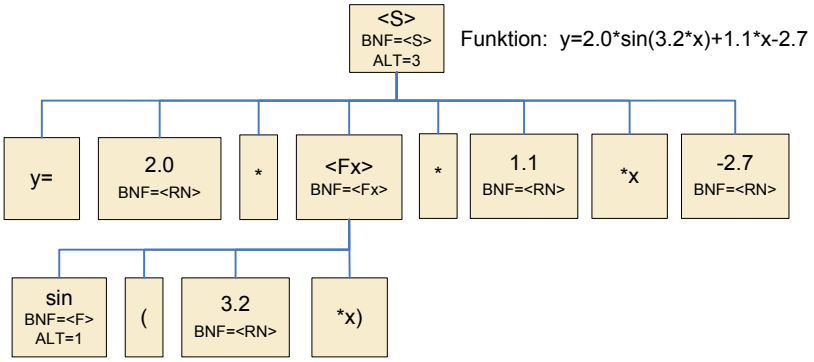


Abbildung 4-22: Extremfall vollständiger Mutation ab dem Startsymbol

4.2.6.2 Probabilistische Mutation

Eine weitere Form der Mutation, die im Rahmen dieser Arbeit entwickelt wurde, ist die probabilistische Mutation. Diese basiert zunächst auf dem üblichen Mutationsverfahren reell codierter genetischer Algorithmen [POH99]. Reelle und ganzzahlige Variablen werden hierbei durch Addition bzw. Subtraktion geringer Werte zum eigentlichen Zahlenwert verändert. Der zu addierende Betrag wird als *Mutationsschritt* bezeichnet. Die Variable wird durch Mutation nicht neu initialisiert, sondern lediglich geringfügig variiert. Das entspricht wesentlich eher dem ursprünglichen Gedanken der Mutation, geringfügige Änderungen an der genetischen Information vorzunehmen.

Bestimmung der Mutationsschritte

Die größte Schwierigkeit bei der Mutation reeller Zahlen ist die Festlegung geeigneter Mutationsschritte. Diese sind unter anderem abhängig von der Problemstellung, dem Definitionsbereich einer Variablen, dem Optimierungsverlauf sowie weiteren zu bestimmenden Einflüssen. Somit ist die Bestimmung einer einzigen Schrittweite nicht ohne weiteres möglich. Im Allgemeinen gelten kleine Schrittweiten als zielführend, jedoch können auch teilweise größere Schrittweiten sinnvoll sein und beispielsweise zur Überwindung lokaler Extrema führen. Daher bietet sich an, die Schrittweite durch Zufallswerte innerhalb gewisser Grenzen zu bestimmen.

In der Literatur werden neben der linearen Verteilung auch unterschiedliche Verteilungsfunktionen propagiert, beispielsweise die Exponentialverteilung nach [POH99] (4-4).

Mutationsschrittweite: $$Var_i^{Mut} = Var_i + s_i \cdot 2^{-u*k} \quad \textbf{(4-4)}$$

Richtung der Mutation (negativ / positiv): $$s_i \in \{-1,1\}$$

gleichverteilter Zufallswert: $$u \in [0,1]$$

Mutationspräzision: $$k \in [>0,\infty]$$

Ein günstigerer Ansatz, der für die Anwendung im Rahmen der beschriebenen Optimierungskomponente gewählt wurde, ist eine probabilistische Verteilung, abgeleitet von der inversen Gaußschen Normalverteilung (4-5).

Probabilistische Mutationsschrittweite:

$$Var_i^{Mut} = Var_i + s_i \cdot \sqrt{-2\ln(u)} \quad \textbf{(4-5)}$$

Richtung der Mutation (negativ / positiv):

$$s_i \in \{-1,1\}$$

gleichverteilter Zufallswert:

$$u \in [0,1]$$

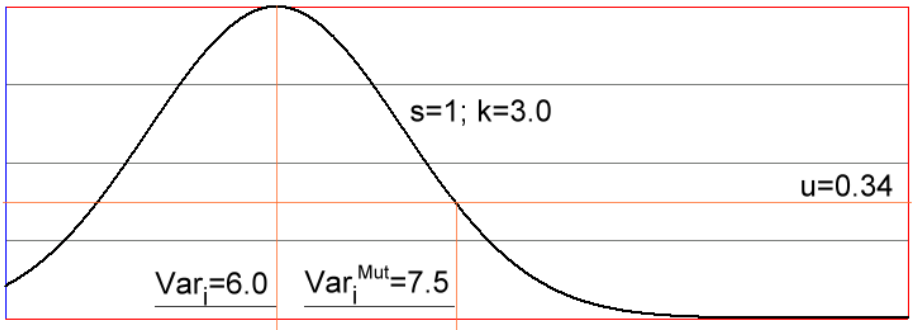


Abbildung 4-23: Probabilistische Mutation

Der beschriebene Mutationsoperator lässt sich prinzipiell auf alle reell-codierten und diskreten numerischen Variablen anwenden. Die Übertragung dieses Operators auf die Repräsentationsform der genetischen Programmierung ist in beschränktem Umfang möglich. Da die Mutation in der genetischen Programmierung prinzipiell auf einzelne selektierte Knoten angewendet wird, muss die Anwendung der probabilistischen Mutation auf zwei Typen von Knoten beschränkt sein:

1. Wertebereich-Symbole
2. Geordnete, nicht-terminierte Symbole

Anwendung probabilistischer Mutation auf nicht-terminierte Symbole

Die Anwendung probabilistischer Mutation ist nicht auf Wertebereich-Symbole beschränkt. Für nicht-terminierte Symbole ist die Anwendbarkeit in besonderen Fällen ebenso möglich. Zunächst beinhalten nicht-terminierte Symbole eine Reihe weiterer Symbole, die beliebigen Typs sein können und ihrerseits weitere Symbole enthalten können. Ebenso können eine Reihe alternativer Ausdrücke, die „Alternativen“, enthalten sein. Unabhängig vom Inhalt der untergeordneten Knoten besteht der Informationsgehalt eines nicht-terminierten Symbols aus der gewählten Alternative. Einzig diese Information ist Gegenstand der Mutation, auch wenn im Ergebnis der Mutation Veränderungen der untergeordneten Struktur entstehen.

Die probabilistische Mutation verfolgt das Ziel geringfügiger Änderungen. Eine Änderung kann nur dann als geringfügig oder weit reichend eingestuft werden, wenn sie entsprechend quantifizierbar ist. Das gilt nicht für eine ungeordnete Sammlung gleichwertiger Alternativen. Auch wenn es bei Definition der BNF keine Möglichkeit gibt, Alternativen wirklich ungeordnet (ohne Angabe einer Reihenfolge) zu definieren,

ist die Nachbarschaft zweier Alternativen einer Reihe gleichwertiger Alternativen absolut zufällig. Jede Mutation von einer bestehenden zu einer beliebigen anderen Alternative wäre gleichwertig und somit nicht bewertbar. Beispielsweise ist dies der Fall bei einem Symbol, das die mathematischen Funktionen „sin“, „cos“ und „tan“ repräsentiert. Eine Mutation von „sin“ nach „tan“ wäre in der Stärke ihrer Auswirkung nicht höher zu bewerten als eine Mutation von „sin“ nach „cos“.

Anders verhält es sich bei tatsächlich geordneten Reihen von Alternativen. Hier kann sehr wohl eine probabilistische Mutation erfolgen. Dazu ist es jedoch von essentieller Bedeutung, das nicht-terminierte Symbol als „geordnet“ zu kennzeichnen, um diesen Sachverhalt klarzustellen. Aus diesem Grund ist das Mutationsgewicht der probabilistischen Mutation für nicht-terminierte Symbole nur dann von Bedeutung, wenn diese als „geordnet“ gekennzeichnet wurden. Andernfalls kann ein nicht-terminiertes Symbol lediglich zur regulären Mutation herangezogen werden.

```
<G>:=<A0>|<A1>|<A2>|<A3>|<A4>|<A5>|<A6>|<A7>  ;ordered
<NG>:=sin|cos|tan|cot|log|ln|atan|tanh|cosh
```

Listing 4-8: Definition geordneter und ungeordneter NT-Symbole

Zur Anwendung der probabilistischen Mutation auf geordnete, nicht-terminierte Symbole werden deren Alternativen als ganzzahlig diskretisiert angesehen. Einer bestehenden Alternative, die mutiert werden soll, wird ein ganzzahliger Wert zugrunde gelegt, der analog zum Vorgehen bei Wertebereich-Symbolen mutiert wird. Durch Diskretisierung des mutierten Wertes wird eine neue, wahrscheinlich benachbarte Alternative gewählt.

Beispiele für geordnete und ungeordnete nicht-terminierte Symbole sind in Listing 4-8 dargestellt. Abbildung 4-24 zeigt die Anwendung probabilistischer Mutation auf das geordnete Symbol <G>. In diesem Beispiel wird die ursprüngliche Alternative <A4> mittels probabilistischer Mutation zur benachbarten Alternative <A6> verändert.

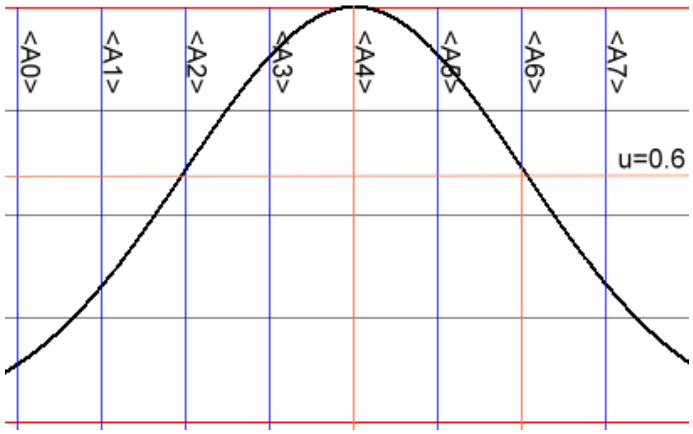


Abbildung 4-24: Probabilistische Mutation nicht-terminierter Symbole

Sicherung bestehender Substrukturen

Die probabilistische Mutation nicht-terminierter Knoten berücksichtigt die Theorie geringfügiger Änderungen auf Ebene eines einzelnen zu mutierenden Symbols. Die den Alternativen untergeordneten Knoten bleiben hierbei jedoch zunächst unberücksichtigt. Gerade hier sind jedoch weit reichende Änderungen zu erwarten, wenn nach einer Mutation einer Alternative sämtliche untergeordneten Knoten (wie bei regulärer Mutation) neu initialisiert würden.

```
<S>:="y="<FX>
<FX>:=<F0>|<F1>|<F2>|<F3>  ;ordered
<F0>:=<RN>
<F1>:=<RN>+x*<RN>
<F2>:=<RN>+x*<RN>+x^2*<RN>
<F3>:=<RN>+x*<RN>+x^2*<RN>+x^3*<RN>
<RN>:=[-5;10;0.1]
```

Listing 4-9: Bildungsvorschrift mit zu sichernden Substrukturen

Benachbarte Alternativen eines nicht-terminierten Knotens weisen in ihren Substrukturen häufig gleiche Knotentypen in ähnlicher Anzahl auf, die von einer probabilistischen Mutation des übergeordneten Knotens möglichst wenig betroffen sein sollten. Oftmals handelt es sich bei dem Wechsel einer Alternative zur Nächsten lediglich um Ergänzungen oder Kürzungen der gleichen Grundinformation. Aus dieser Erkenntnis wurde ein Mechanismus entwickelt, der bei probabilistischer Mutation nicht-terminierter Symbole möglichst viele Bestandsknoten in die neue Alternative übernimmt. Lediglich die zuvor nicht vorhandenen Informationen werden zufällig initialisiert. Umgekehrt wird im Fall überflüssiger Knoten eine zufällige Auswahl getroffen.

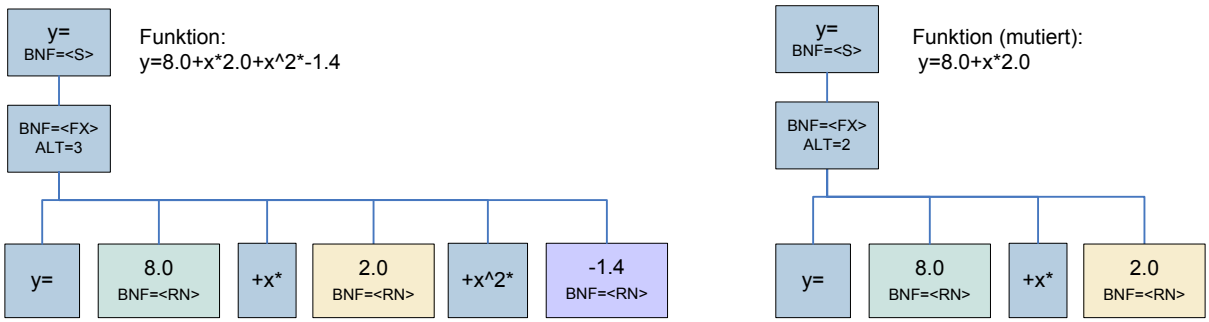


Abbildung 4-25: Probabilistische Mutation unter Beibehaltung bestehender Substrukturen

Eine mögliche Bildungsvorschrift zur Verdeutlichung der beschriebenen Problematik ist in Listing 4-9 gegeben. Ein auf dieser Bildungsvorschrift basierendes Individuum ist in Abbildung 4-25 (links) dargestellt. Eine probabilistische Mutation dieses Individuums an Knoten <FX> von Alternative 2 auf 1 würde ohne den beschriebenen Mechanismus eine vollständige Reinitialisierung sämtlicher untergeordneter Knoten bewirken. Die Sicherung von Substrukturen bewirkt, dass die vor der Mutation vor-

handenen Knoten des Typs <RN> in das mutierte Individuum in erforderlicher Anzahl und Reihenfolge übertragen werden, wie in Abbildung 4-25 (rechts) zu erkennen.

4.2.7 Bloat

Ein bekanntes Phänomen der genetischen Programmierung ist das so genannte *Code-Growing*[3] oder auch *Bloat*[4]. Hierbei handelt es sich um die generationsweise anwachsende Tiefe der Struktur eines Genotypen, die nicht wesentlich zur Verbesserung der Fitness beiträgt. Voraussetzung zur Entstehung von Bloat ist die rekursive Definition der Bildungsvorschrift, da ansonsten die maximale Tiefe eines Individuums bereits durch Einhaltung der Bildungsregeln beschränkt wäre. Eine weitere Voraussetzung für Bloat ist die Anwendung entsprechender genetischer Operationen, die das Anwachsen der Individuen ermöglichen.

Die ursprüngliche Definition der genetischen Programmierung nach [KOZ92] erlaubt beispielsweise für die Rekombination lediglich die Kreuzung zweier Individuen an Knoten, die sich an der gleichen Stelle befanden. Somit konnte alleine durch Rekombination keinerlei Zuwachs der Hierarchietiefe eines Individuums erreicht werden, sondern nur durch Mutation. Bei Anwendung des grammatikbasierten GP-Ansatzes der vorliegenden Arbeit wäre Bloat infolge Rekombination durchaus denkbar, da zur Kreuzung zweier Knoten lediglich deren Typ identisch sein muss, nicht jedoch die Position innerhalb der Individuen.

Die Gründe für das Entstehen von Bloat sind vielfältig, und es existieren mehrere Theorien dazu. Die wahrscheinlich älteste Theorie [LAN02], [TAC94] besagt, dass der genetische Algorithmus versucht, sinnvollen Code durch Einführung von wertlosem Code vor Rekombination zu schützen. Dies ist einleuchtend, da mit steigender Anzahl „wertloser“ Knoten die Wahrscheinlichkeit der Rekombination an diesen zunimmt, was faktisch zu keiner Veränderung des Individuums führt.

Zur Vermeidung von Bloat sind beispielsweise folgende Strategien sinnvoll anwendbar:

- Verzicht auf Rekursion
- Einsatz von Straffunktionen basierend auf der Codegroße zur Beeinflussung der Fitness
- Bereinigung der Individuen um wertlosen Code
- Beschränkung der Hierarchietiefe
- Beschränkung der Anzahl rekursiver Knotentypen

Bei den im Rahmen der vorliegenden Arbeit erarbeiteten Anwendungsbeispielen konnte die Abbildung der Bildungsvorschrift ohne Rekursion erfolgen, daher tritt Bloat in den gezeigten Beispielen nicht auf. Die vorgestellte GP-Optimierungskomponente ist jedoch universell einsetzbar, weshalb Rekursion für andere Einsatzgebiete nicht grundsätzlich ausgeschlossen werden kann. Daher enthält die Bildungsvorschrift die Möglichkeit der Definition von Regeln zur Beschränkung der Strukturtiefe. Ebenso lassen sich einzelne Konten anhand ihres Typs oder weiterer Merkmale beschrän-

[3] Code-growing: Anwachsender Programmcode

[4] Bloat: Aufblähen (der Individuen)

ken. Weitere Mechanismen, wie beispielsweise Straffunktionen, können bei Bedarf jederzeit durch die externe Fitnessfunktion innerhalb der Wissensbank ergänzt werden, und sind daher nicht originärer Bestandteil der entwickelten Bildungsvorschrift.

4.2.8 Parameteradaption

Der Optimierungsprozess wird durch eine Reihe globaler Optimierungsparameter gesteuert. Die Wichtigsten hiervon sind die Populationsgröße sowie die Rekombinations- und Mutationsraten. Die Wahl der Optimierungsparameter trägt wesentlich zum Verhalten der genetischen Suche und zu deren Performanz bei. Wie in Kap. 2.1.2.5 erläutert, basiert die genetische Suche im Wesentlichen auf den Mechanismen "Exploration" und "Exploitation", deren Verhältnis indirekt durch Anpassung der Optimierungsparameter gesteuert werden kann.

In der Anfangsphase des Optimierungsprozesses liegen noch keinerlei Informationen über den Suchraum vor. Daher ist es zu Beginn sinnvoll, den Mechanismus der Exploration zu präferieren, um eine breite Abdeckung des Suchraumes zu erreichen. Im weiteren Verlauf der Optimierung sollten sich Individuen im Umfeld lokaler Optima herausbilden. Die detailliertere Verbesserung dieser Individuen innerhalb ihrer Optima, mit dem Ziel der Findung eines globalen Optimums, ist Aufgabe der Exploitation. Der Übergang des Schwerpunktes von Exploration zu Exploitation kann fließend sein. Ebenso kann es unter bestimmten Umständen sinnvoll sein, während des Optimierungsprozesses zwischenzeitlich wieder zur Exploration zurückzukehren, um einer in lokalen Optima „gefangenen“ Population den Suchraum weiter erkunden zu lassen.

Im Rahmen der vorliegenden Arbeit wurde zur generationsabhängigen Adaption der Optimierungsparameter ein dreistufiger Ansatz gewählt (Abbildung 4-26). In der ersten Stufe werden entsprechend des Gedankens der Exploration genetische Operationen mit stärkeren Beeinflussungen der Individuen bevorzugt. Hierdurch wird primär eine hohe Diversität erreicht, jedoch ist in dieser Phase die Fähigkeit detaillierter Optimierung von bereits als relativ gut bewerteten Individuen beschränkt. Das hierfür erforderliche Verhalten wird durch die Wahl der Parameter in der dritten Phase erreicht. Hierbei werden genetische Operationen favorisiert, die nur geringe Änderungen an den Individuen vornehmen, um so eine schrittweise Annäherung an das tatsächliche Optimum zu ermöglichen. Die zweite Phase beschreibt für die meisten Parameter einen linearen Übergang von der ersten zur dritten Phase. Parameter, für die ein linearer Übergang nicht sinnvoll erscheint, erfahren eine sprunghafte Änderung zu Beginn oder Ende der Übergangsphase.

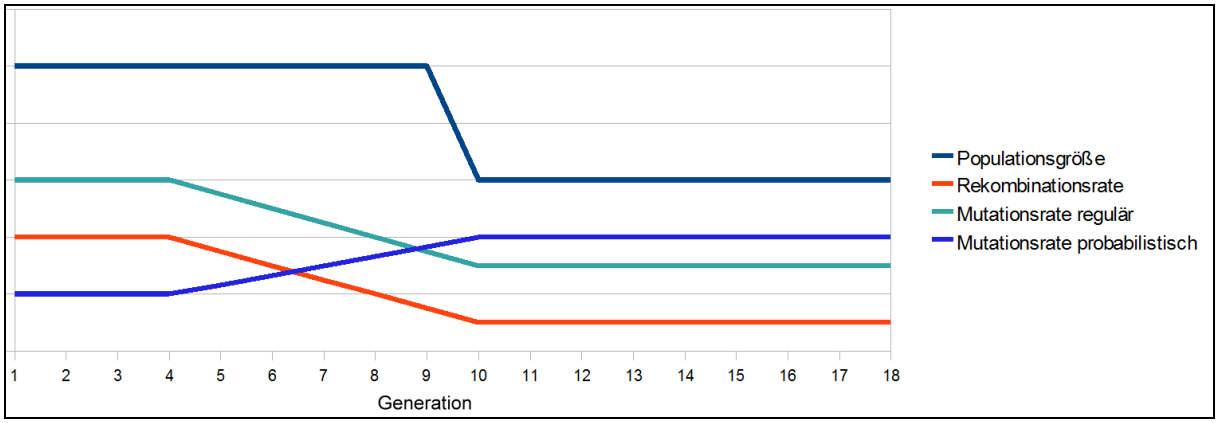


Abbildung 4-26: Parameteradaption in Abhängigkeit des Generationsfortschrittes

4.2.9 Komponente zur Definition der Bildungsvorschrift

Ähnlich der in Kap. 4.1.5 beschriebenen Wissensakquisitionskomponente wurde zur Definition und Bearbeitung der Bildungsvorschrift in Form der BNF ein Werkzeug erstellt, der „GPCore-Manager“ (Abbildung 4-27). Dieses Werkzeug erlaubt die Definition der BNF-Grammatik wahlweise in textbasierter Form oder durch dialogbasierte Eingaben. Weiterhin wird die komfortable Bestimmung detaillierter, knotenspezifischer Parameter unterstützt.

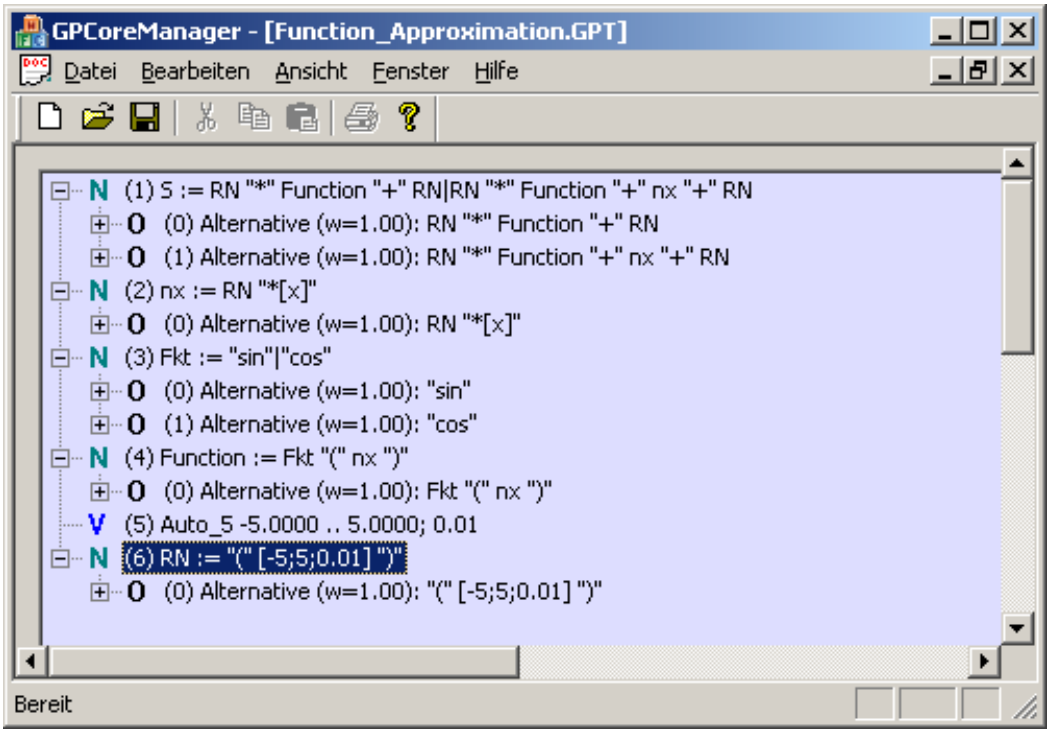


Abbildung 4-27: GPCore-Manager (Ausschnitt)

Der GPCore-Manager verfügt weiterhin über eine Testumgebung, die zur Überprüfung der Bildungsvorschrift sowie des Optimierungsverhaltens eingesetzt werden kann (Abbildung 4-28). Hier besteht die Möglichkeit der zufälligen Initialisierung einzelner Individuen, schrittweiser Test des Rekombinations- und Mutationsverhaltens sowie Ausgabe des erzeugten Programmcodes einzelner Individuen. Zu Demonstrationszwecken besitzt die Testumgebung einen Optimierungsteil, der exemplarisch die Verarbeitung mathematischer Funktionen unterstützt.

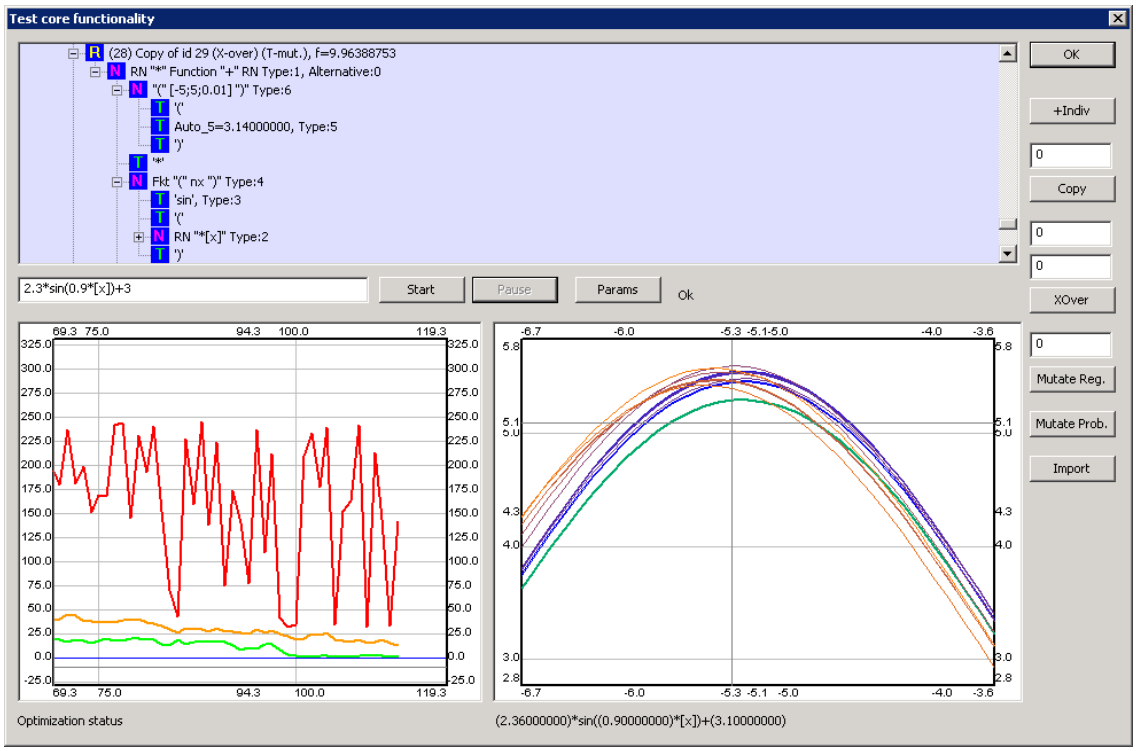


Abbildung 4-28: Testumgebung

4.2.10 Software-Design

Die in Abbildung 4-29 dargestellte Klassenstruktur lässt drei größere Einheiten erkennen:

- die GP-Kernkomponente
- Klassen zur Repräsentation der Grammatikdefinition
- Klassen, die der Abbildung der Individuen dienen

Die Klasse GP_Optimization stellt die Hauptklasse der Kernkomponente dar. Sie enthält die globalen Optimierungsparameter, Verweise auf eine Instanz der BNF-Grammatikdefinition *GP_BNF_Root* sowie eine Liste parallel zu führender Populationen, die üblicherweise nur eine einzelne Instanz enthält. Jede dieser Populationen beinhaltet eine Liste von Generationen, die während der Optimierung um Instanzen neuer Generationen ergänzt wird. Aus Gründen der Nachvollziehbarkeit können vorangegangene Generationen vollständig erhalten bleiben. Alternativ erfolgt nach Erstellen einer neuen Generation eine automatische Bereinigung der vorangegangenen Generationen. Eine Generation enthält eine Liste ihrer Individuen, die durch Instanzen des Root-Symbols GP_IND_Root repräsentiert werden.

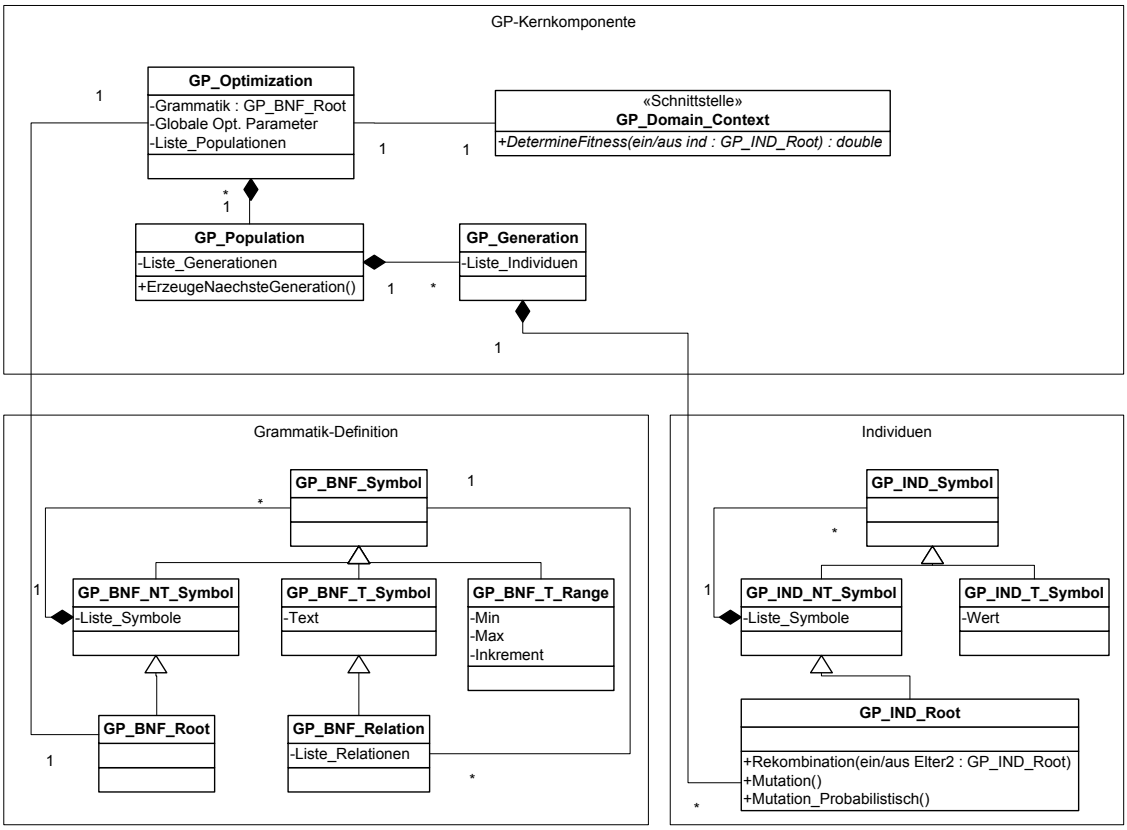


Abbildung 4-29: Vereinfachte Klassenstruktur der Optimierungskomponente

5 Implementierung des Entwurfssystems

Zur Evaluation der entwickelten Ansätze wurde das prototypische Entwurfssystem „Evolutionary DaVinci“ als modularisierte Software-Applikation implementiert. Kern des Entwurfssystems ist die Produktmodellierung zur Repräsentation von zu optimierenden Bauwerken in den zur Simulation und Bewertung erforderlichen Entwurfsphasen. Neben dem Bauwerksmodell integriert das Entwurfssystem die wissensbasierte Plattform (Kap. 4.1) sowie die Optimierungskomponente (Kap. 4.2).

Aus Sicht des Anwenders bietet das Entwurfssystem eine einheitliche Umgebung zur zentralen Verwaltung und Bearbeitung sämtlicher Vorgaben und Randbedingungen, sowie zur Definition und Pflege der Wissensbanken. Ebenso stellt die Plattform die Laufzeitumgebung für den selbständig ablaufenden Optimierungsprozess bereit. Sowohl während als auch nach Beendigung eines Optimierungsvorgangs besteht die Möglichkeit, einzelne Individuen zu visualisieren sowie anhand der wissensbasierten Erklärungskomponente detaillierte Parameter und Berechnungsergebnisse innerhalb der simulierten Modelle einzusehen. Während des automatisierten Optimierungsprozesses ist ausschließlich die Optimierungskomponente für die Steuerung und Kontrolle des Gesamtprozesses verantwortlich.

5.1 Produktmodellierung

Die Erstellung eines parametrisierten Tragwerksmodells ist ein mehrstufiger Prozess. Daher erfolgt die objektorientierte Modellierung der Tragstruktur in drei Teilmodellen unterschiedlichen Detaillierungsgrades. Ein wichtiger Aspekt ist die Integration wissensbasierter Modellierung des Strukturmodells, so dass dessen Ausprägung möglichst variabel durch den Anwender adaptierbar ist. Das im Rahmen dieser Arbeit entwickelte System basiert zunächst auf einer Reihe von Annahmen, die im Folgenden erläutert werden.

Grundlegender Ansatz dieser Arbeit ist die Top-Down-Modellierung. Die Planung erfolgt schrittweise von relativ groben Spezifikationen hin zur immer detaillierteren Planungsphasen. Ein Planer beginnt einen klassischen Entwurfsprozess üblicherweise nicht mit dem detaillierten Entwurf einzelner Bauteile, sondern er stellt zunächst abstraktere Überlegungen an. Diese umfassen die Dimensionierung der äußeren Gebäudehülle, Festlegung von Grundfläche, Geschossanzahl, etc. In weiteren Schritten erfolgt der Entwurf der Grundrissbegrenzung, woran sich die Anordnung der Gebäudeachsen orientiert.

Ein großer Teil der Grundrissgestaltung beschränkt sich beim Entwurf von Hochhäusern, insbesondere bei rechtwinkligen Grundrissen, auf die Wahl der Anzahl und Abstände der Gebäudeachsen. An deren Kreuzungspunkten werden als vertikale Tragglieder meist Stützen ausgebildet. Weiterhin erfolgt die Festlegung von vertikalen Erschließungsbereichen in Form von Kernen, die sich teilweise oder vollständig über alle Stockwerke erstrecken. Bauteile wie Unterzüge und Wände orientieren sich üblicherweise an dem vorgegebenen Gebäuderaster.

Die einzelnen Stockwerke bilden in den meisten Fällen einen weitgehend einheitlichen Grundriss, der in einem Teil der Stockwerke um aussteifende Elemente ergänzt werden kann. In oberen Geschossen kann häufig eine Reduktion der Querschnitte vertikaler Tragglieder oder auch der Entfall eines Teils der aussteifenden Elemente erfolgen. Eine Ausnahme bilden Geschosse mit Outriggern bzw. abweichender Nutzungsart, beispielsweise Technikgeschosse [LAU04]. Aus verschiedenen Gründen werden häufig Geometrien in den Stockwerken gefordert, die einer wirtschaftlichen

Gestaltung des Tragsystems entgegenstehen. Es ist unüblich, jedes Stockwerk eines Hochhauses unterschiedlich auszubilden. Vereinfachend wird im Rahmen der vorliegenden Arbeit davon ausgegangen, dass zumindest zusammenhängende Gruppen von Stockwerken identisch ausgeprägt werden. Im Fall eines Technikgeschosses kann eine Gruppe auch hilfsweise aus einem einzelnen Stockwerk bestehen.

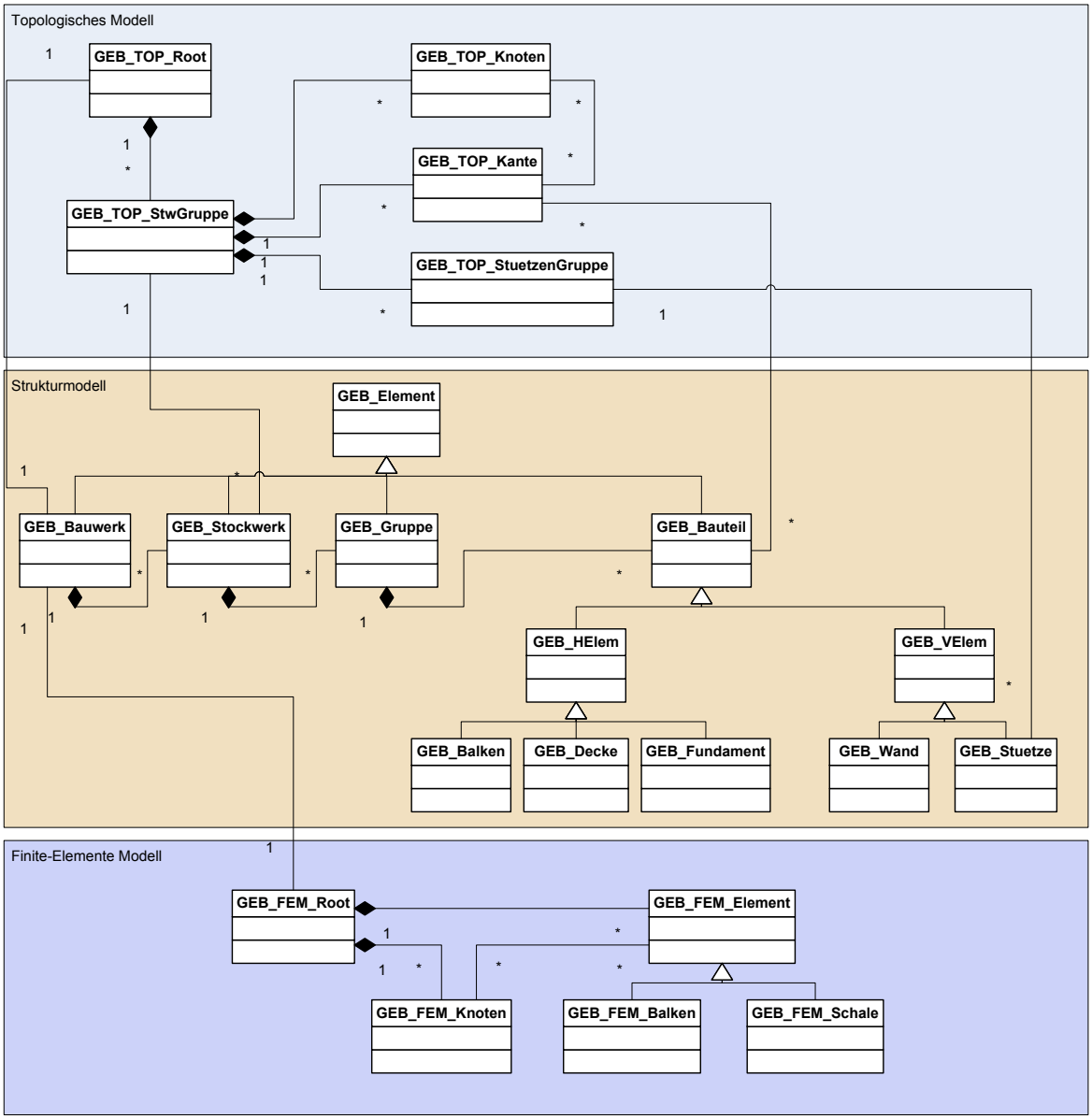


Abbildung 5-1: Auszug aus der Klassenstruktur des Produktmodells

Das Produktmodell gliedert sich in ein grobes Topologiemodell, ein detaillierteres Strukturmodell und ein feingranulares Finite-Elemente-Modell. Die Modelle sind voneinander abhängig und werden zur Simulation eines Bauwerks während des automatisierten Bewertungsprozesses schrittweise aufgebaut. Abbildung 5-1 zeigt einen Überblick über die Produktmodellierung mit der beschriebenen Gliederung.

5.1.1 Topologie

Basierend auf einer Reihe von Entwurfsparametern, die jeweils aus Informationen des genetischen Codes eines Individuums oder zugehörigen wissensbasierten Berechnungen gewonnen werden, erfolgt die Erstellung eines parametrisierten topologischen Modells. Das topologische Modell dient der ersten, groben Strukturierung des Tragwerkmodells. Es bildet die Grundlage der nachfolgenden Modellierungsphasen, die ihre Elemente an den Elementen des topologischen Modells orientieren. Im Einzelnen kommen folgende Entwurfsparameter zum Einsatz:

- Anzahl und Abstand der Gebäudeachsen, jeweils für X- und Y-Richtung
- Eine beliebige Anzahl von Stockwerkgruppen
- Je Stockwerkgruppe die Anzahl der Stockwerke sowie die Stockwerkhöhe

Aus den gegebenen Parametern lässt sich ein dreidimensionales räumliches Strukturnetz generieren. Dieses Netz wird durch seine Knoten und Kanten definiert. Hierbei wird eine Kante jeweils nur von einem Knoten zum nächsten gebildet, so dass keine achsen- oder stockwerkübergreifenden Kanten entstehen. Beispielhaft sind in Listing 5-1 die notwendigen Parameter zur Erstellung eines Topologiemodells aufgeführt.

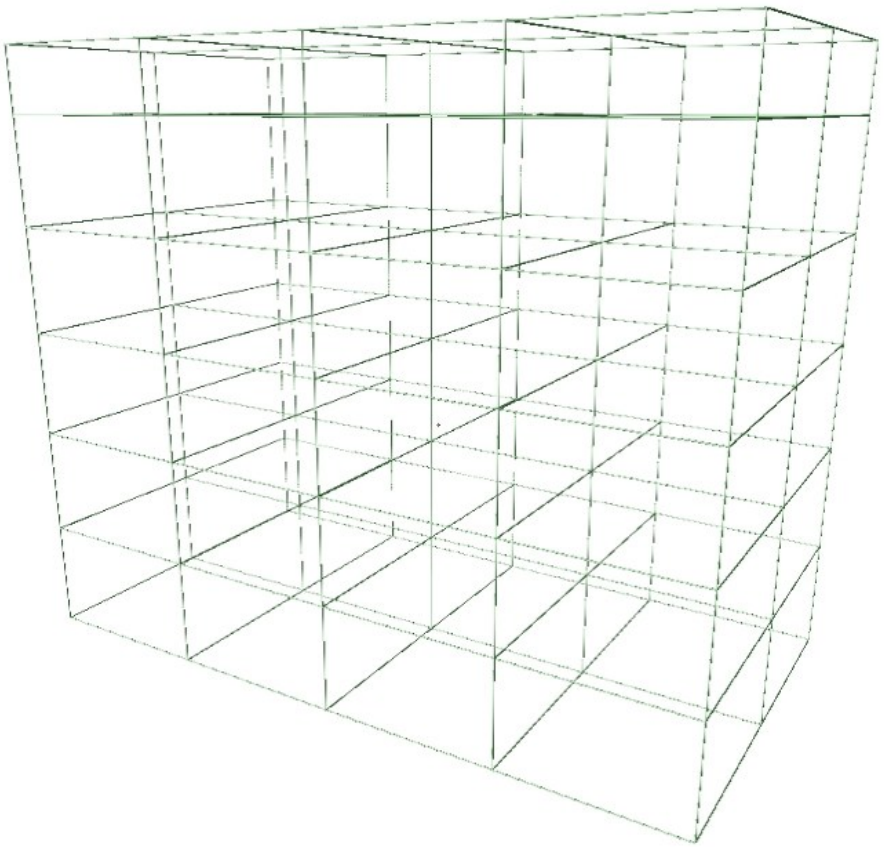


Abbildung 5-2: Topologiemodell gemäß Listing 5-1

Die grafische Darstellung des zugehörigen Topologienetzes ist in Abbildung 5-2 gegeben. Dieses Beispiel verwendet einen einfachen Rechteckgrundriss mit regelmäßigem Stützenraster. Ebenso sind theoretisch andere Grundrissformen möglich, beispielsweise L-förmige, runde oder zusammengesetzte Formen. Je nach Ausprägung der Grundrisse können möglicherweise weitere Parameter erforderlich sein.

```
Bauwerk:
Grundrisstyp=Rechteck

Raster:
Typ=regelmäßig, X=8,30m, Y=8,80m, Felder_X=4, Felder_Y=2

Stochwerkgruppe 1:
Anzahl_Stockwerke=5, Stockwerkhöhe=5,00m

Stochwerkgruppe 2:
Anzahl Stockwerke=0 (alle Restlichen des Gebäudes), Stock-
werkhöhe=3,00m
Outrigger=J
...
```

Listing 5-1: Parameter zur Erzeugung eines Topologiemodells

5.1.2 Strukturmodell

Das Strukturmodell beschreibt die einzelnen Strukturobjekte (Bauteile) des Tragwerks. Die Erstellung der Bauteile erfolgt vollständig wissensbasiert durch Auswertung von Design-Selektionen an sämtlichen Kanten des Topologiemodells. Durch die in der Design-Wissensbank formalisierten Regeln wird für jede Kante anhand ihrer Raumorientierung, Lage im Bauwerk, zugehörigen Stockwerkgruppe und benachbarten aussteifenden Elementen entschieden, ob und welche Bauteile in welchen Dimensionen erzeugt werden. Dem Anwender bleibt es somit beispielsweise überlassen, wie die Beschreibung aussteifender Bauteile und zusammengesetzter Querschnitte in der Bildungsvorschrift definiert und später durch wissensbasierte Regeln interpretiert und ausgeführt wird.

Sämtliche Bauteile orientieren sich an einer Systemachse, die nicht notwendigerweise innerhalb des jeweiligen Bauteils verlaufen muss. Die Systemachse wird normalerweise mit einer Kante des Topologiemodells zur Deckung gebracht. Zusätzliche Abstandsparameter (Offsets) bestimmen die Abstände der Bauteilgeometrie zu den Endknoten ihrer Systemachse. So ist beispielsweise der Volumenkörper einer Stütze um die beiden umgebenden Deckenstärken kürzer als ihre Systemachse. Die Systemachse eines Unterzugs liegt in der Mitte der darüber befindlichen Decke. Neben der Bauteilposition und -dimension erfolgt auch die Ermittlung der Offsets durch wissensbasierte Berechnungen und ist somit anwenderseits anpassbar.

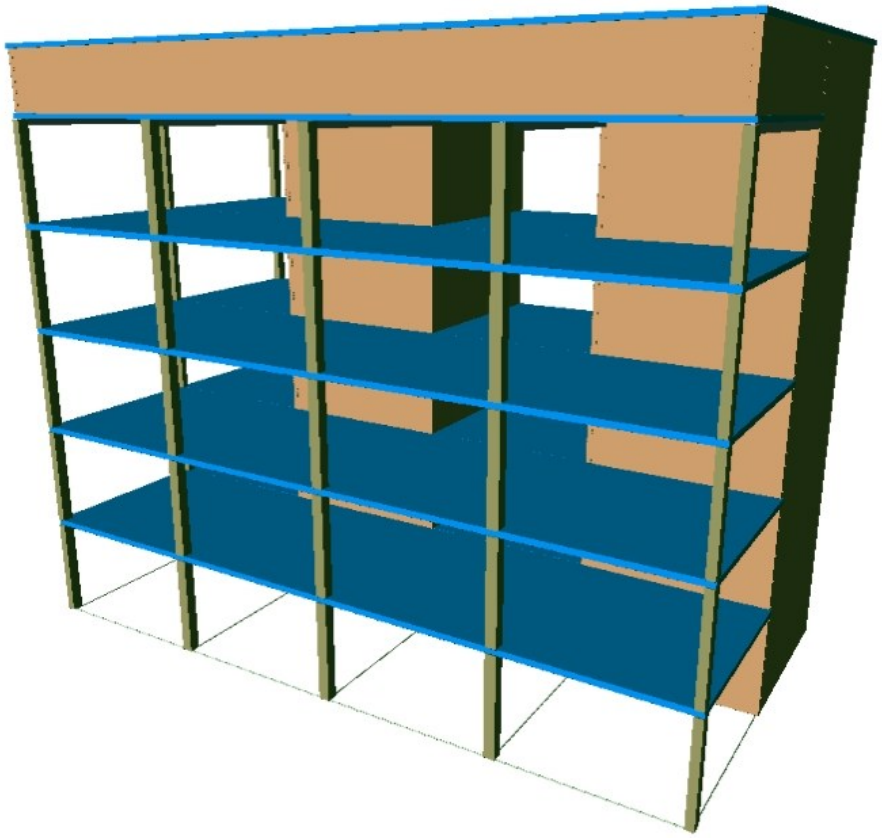


Abbildung 5-3: Strukturmodell mit zwei Kernen und einem Outriggergeschoss

5.1.3 Finite-Elemente-Modell

Die mit dem Entwurfssystem simulierten Tragwerksmodelle werden durch das externe System B&B in dynamischer und statischer Hinsicht berechnet. Zur Vorbereitung der externen Berechnung ist es notwendig, das Strukturmodell in ein detailliertes FE-Modell zu überführen. Die hierfür erforderlichen Klassen sind als Teilmodell in Abbildung 5-1 dargestellt.

5.2 Beschreibungssprache für Tragwerksmodelle

Die zur Optimierung eines Tragwerks erforderlichen Varianten bedürfen einer formalen Beschreibungssprache, die eine eindeutige Formalisierung sämtlicher im Rahmen der Optimierung erforderlichen Varianten ermöglicht. Ein auf dieser Beschreibungssprache basierender Text enthält eine Reihe eindeutiger Handlungsanweisungen und kann somit als eine Art „Programmcode“ angesehen werden. Durch Anwendung eines jeden korrekt formulierten Programmcodes sollte sich immer ein eindeutiges Tragwerk erzeugen lassen. Der Programmcode entspricht in der Optimierung dem Genotypen eines Individuums, das daraus erzeugte Strukturmodell entspricht dem Phänotypen. Die Beschreibungssprache bildet die Grundlage der Bildungsvorschrift.

Die syntaktische und semantische Definition der Bildungsvorschrift erfolgt mittels der im Rahmen dieser Arbeit erweiterten Variante der Backus-Naur Form. Somit ist gewährleistet, dass die Optimierungskomponente einen sprachlich korrekten Formalismus zur Erzeugung von Individuen verwendet. In Listing 5-2 ist eine einfache Bildungsvorschrift zur Beschreibung von Tragsystemen gegeben, wie sie auch für die

weiteren Experimente dieser Arbeit Verwendung findet. Diese Definition ist lediglich ein Beispiel zur Demonstration der Möglichkeiten, die das entwickelte Entwurfssystem bietet, ohne den Anwender auf diese Definition zu beschränken.

Die Syntax sieht vor, dass eine hierarchische Gliederung der vorliegenden Informationen erfolgt. Zusammengehörige Elemente werden durch eckige Klammern eingeschlossen, die eine Sammlung von Attributen sowie weitere untergeordnete Gruppen beinhalten können. Das Beispiel beginnt mit einem Startsymbol <S>, das grundsätzlich ein Gebäude *GEB* definiert. Dieses Gebäude stellt eine Komposition aus den Bereichen „Grundriss“, „Kerne“ und „Stockwerkgruppen“ dar.

Der Bereich „Grundriss“ definiert Parameter zur Gestaltung des topologischen Modells. Hier sind eine Reihe von Grundformen und Kombinationen denkbar. In der prototypischen Implementierung dieser Arbeit werden derzeit regelmäßige Rechteckgrundrisse unterstützt, daher bietet der Grundriss an dieser Stelle noch keine Alternativen. Weiterhin enthalten sind Informationen über den Aufbau des Rasters. Je Achse werden hier die Anzahl der Felder sowie der Achsabstand angegeben. Alternativ könnte statt des Abstandes auch die Gebäudebreite vorgegeben und der jeweilige Achsabstand durch eine Formel der Wissensbank ermittelt werden. Eine solche Anpassung kann jederzeit durch den Anwender selbst vorgenommen werden.

Der Bereich „Kerne“ vereint Informationen über einen oder mehrere zu Gebäudekernen zusammengesetzte Querschnitte. An dieser Stelle wird bewusst auf eine rekursive Notation verzichtet, da sich ansonsten im Laufe der Optimierung schnell Bloat (siehe Kap. 4.2.7) einstellt. Die vorliegende Definition ermöglicht explizit ein bis vier Kerne, was je nach Komplexität der Tragstruktur angepasst werden kann. Die Informationen eines einzelnen „Kerns“ beschränken sich auf Lage, Dimension, Wandstärke und maximale Stockwerkshöhe. Sowohl Lage als auch Dimension sind relativ im prozentualen Verhältnis zum gegebenen Raster definiert.

Die Transformation zwischen der relativen und der absoluten Position innerhalb des Topologierasters erfolgt anhand wissensbasiert formalisierter Funktionen. Dieses Vorgehen ist vorteilhaft, da bei Definition der Positionsangaben keine Aussage über die tatsächliche Anzahl der Gebäudeachsen getroffen werden kann. Gleichzeitig wird die Übertragung genetisch wertvoller Informationen zwischen unterschiedlichen Individuen ermöglicht. Somit kann die Information, dass Tragwerk „A“ bei X = 50 % einen Kern besitzt, problemlos auf Tragwerk „B“ übertragen werden, auch wenn die topologischen Modelle beispielsweise für Tragwerk „A“ 13 Achsen und für Tragwerk „B“ nur 7 Achsen in X-Richtung vorsehen.

```
<S>              :=   <Bauwerk>
<Bauwerk>        :=   GEB[<Grundriss> <Kerne> <SWgruppen>]
<Grundriss>      :=   GRUNDRISS=RECHTECK; <Raster>
<Raster>         :=   RASTER=REGEL; FELDER_X= <Fel-
                      der>;FELDER_Y=<Felder>;RASTER_X=<Rastergroesse>;RASTE
                      R_Y=<Rastergroesse>
<Felder>         :=   [2;5;1]
<Rastergroesse>  :=   [5;9;0.05]  (Raster zwischen 5 und 9m, Schrittweite
                      5cm)
<Kerne>          :=   <Kern>|<Kern> <Kern>|<Kern> <Kern> <Kern>|<Kern>
                      <Kern> <Kern> <Kern>
<Kern>           :=   Kern[KFELD_X_PRZ=<KernfeldX>;KFELD_Y_PRZ=
                      <KernfeldY>;KFELDER_X_PRZ=<KernfelderX>;
                      KFELDER_Y_PRZ=<KernfelderY>; KMAX_STW=<KMAX_STW>]
<KernfeldX>      :=   [0;1;0.01]  (Kernposition zwischen 0% und 100%,
```

```
                        Schrittweite 1%)
<KernfeldY>      :=     [0;1;0.01]
<KernfelderX>    :=     [0;0.4;0.01]  (Kernbreite zwischen 0% und 40% der
                        Gebäudebreite, Schrittweite 1%)
<KernfelderY>    :=     [0;0.4;0.01]
<KMAX_STW>       :=     [2;20;1]
<SWgruppen>      :=     <SWgruppe_Ende>|<SWgruppe> <SWgruppe_Ende>|
                        <SWgruppe> <SWgruppe_Outr> <SWgruppe> <SWgrup-
                        pe_Ende>|<SWgruppe> <SWgruppe_Outr> <SWgruppe>
                        <SWgruppe_Outr> <SWgruppe> <SWgruppe_Ende>
<SWgruppe>       :=     SWG[ANZ_STW= [1;6;1];H_STW= [3;5;.1]; OUTRIGGER
                        =N;<Deckensystem> <Ssg1> <Ssg2> <Ssg3> <Ssg4>]
<SWgruppe_Outr>  :=     SWG[ANZ_STW=1;H_STW=[2;4;.2];OUTRIGGER=J;
                        <Deckensystem> <Ssg4>]
<SWgruppe_Ende>  :=     SWG[ANZ_STW=0;H_STW=[3;5;.1]; OUTRIGGER =N; <Decken-
                        system> <Ssg1> <Ssg2> <Ssg3> <Ssg4>]
<Deckensystem>   :=     <Fd>|<Uzd>
<Fd>             :=     DS=FD;H_D=[0.2;0.4;0.01]
<Uzd>            :=     DS=UZD;H_D= [0.2;0.4;0.01];H_UZ=[0.2;0.4;0.01];
                        B_UZ=[0.2;0.6;0.01];DIR_UZ=<Dir_Uzd>
<Dir_Uzd>        :=     x|y
<Ssg1>           :=     SSG[NAME=x-edge;DX=<SsgDim>;DY= <SsgDim>]
<Ssg2>           :=     SSG[NAME=y-edge;DX=<SsgDim>;DY= <SsgDim>]
<Ssg3>           :=     SSG[NAME=corner;DX=<SsgDim>;DY= <SsgDim>]
<Ssg4>           :=     SSG[NAME=inside;DX=<SsgDim>;DY= <SsgDim>]
<SsgDim>         :=     [0.16;0.60;0.1]
```

Listing 5-2: Bildungsvorschrift (vereinfacht)

Die horizontale Gliederung des vereinfachten Tragwerkmodells erfolgt durch Stockwerkgruppen. Die Gruppe der obersten Stockwerke hat in diesem Beispiel eine Sonderrolle, da die Gesamtzahl der Stockwerke als äußere Randbedingung in der Wissensbank vorgegeben wird. Daher enthält die oberste Stockwerkgruppe die Anzahl 0, was als „undefiniert“ interpretiert wird und zur wissensbasierten Ermittlung der notwendigen fehlenden Stockwerke führt. Weiterhin sind gesonderte Stockwerkgruppen, die genau ein Stockwerk enthalten, mit Belt-Truss Outriggern vorgesehen. Die Zusammenstellung einiger sinnvoller Folgen von regulären, Outrigger- und Endgeschossen erfolgt durch das Symbol <SWgruppen>. Die einzelnen Stockwerkgruppen enthalten Informationen über die Anzahl der Stockwerke, die Stockwerkshöhe, gesonderte Aussteifungsmerkmale sowie das Deckensystem und mehrere Stützensystemgruppen.

Das Deckensystem einer Stockwerkgruppe kann im gezeigten Beispiel als Flachdecke oder als Unterzugsdecke in x- oder y-Orientierung ausgebildet werden. Im Fall der Flachdecke ist lediglich die Deckenstärke anzugeben. Die Unterzugsdecke erfordert zusätzlich die Angabe der Orientierung (x/y) sowie die Dimensionierung der Unterzüge. Stützensystemgruppen umfassen jeweils eine Gruppe von Stützen mit ähnlichen Lasteinzugsbereichen, so dass eine gemeinsame Dimensionierung innerhalb einer Stockwerkgruppe sinnvoll sein kann. Weiterhin können Stützensystemgruppen zur Ausbildung von Megastützen an definierten Positionen herangezogen werden. Die Zuordnung einzelner Stützen zu ihrer jeweiligen Systemgruppe erfolgt durch Regeln innerhalb der Wissensbank. Im vorliegenden Beispiel wurde dem System die Freiheit zur Ausbildung ungewöhnlicher Querschnittsverhältnisse gegeben.

In Listing 5-3 ist ein Beispiel für Programmcode (Genotyp) zur Erstellung eines einfachen Bauwerks gegeben.

```
GEB[  GRUNDRISS=RECHTECK;
RASTER=REGEL;
      FELDER_X=5.00000000; FELDER_Y=4.00000000;
      RASTER_X=7.05000000; RASTER_Y=5.70000000;

      KERN[ KFELD_X_PRZ=0.33000000; KFELD_Y_PRZ=0.36000000;
            KFELDER_X_PRZ=0.19000000; KFELDER_Y_PRZ=0.32000000;
            KMAX_STW=10.00000000]

      KERN[ KFELD_X_PRZ=0.18000000;
            KFELD_Y_PRZ=0.66000000;
            KFELDER_X_PRZ=0.12000000;
            KFELDER_Y_PRZ=0.23000000;
            KMAX_STW=9.00000000]

      SWG[  ANZ_STW=2.00000000; H_STW=3.100000000;
            OUTRIGGER=N;
            DS=FD; H_D=0.27000000;
            SSG[  NAME=x-edge; DX=0.58000000; DY=0.19000000]
            SSG[  NAME=y-edge; DX=0.54000000; DY=0.54000000]
            SSG[  NAME=corner; DX=0.17000000; DY=0.28000000]
            SSG[  NAME=inside; DX=0.59000000; DY=0.41000000]]

      SWG[  ANZ_STW=0; H_STW=4.90000000;
            OUTRIGGER=N;
            DS=UZD; H_D=0.37000000; H_UZ=0.20000000;
            B_UZ=0.29000000; DIR_UZ=y;
            SSG[  NAME=x-edge; DX=0.45000000; DY=0.23000000]
            SSG[  NAME=y-edge; DX=0.46000000; DY=0.36000000]
            SSG[  NAME=corner; DX=0.36000000; DY=0.17000000]
            SSG[  NAME=inside; DX=0.41000000; DY=0.17000000]]]
```

Listing 5-3: Genotyp als „Programmcode“ zur Definition eines Bauwerks

5.3 Wissensbasierte Instanziierung des Produktmodells

Die Interpretation des Genotypen erfolgt nicht fest im Quellcode der Software verankert, sondern fast ausschließlich wissensbasiert. Die Design-Wissensbank stellt das Bindeglied zwischen dem gegebenen Genotypen und dem zu erzeugenden Phänotypen dar. Somit entspricht der Programmcode dem Faktenwissen, auf das die Design-Wissensbank angewendet wird. Durch Ergänzung der Grammatik, die dem Genotypen zugrunde liegt, können weitere Optimierungsparameter in den Genotypen einfließen. Die Interpretation dieser Parameter wird daraufhin in der zugehörigen Wissensbank ergänzt und kommt somit bei Erstellung der Tragstruktur zur Anwendung. Ebenso können als gegeben anzusehende oder funktional abhängige Größen aus der Grammatik entfernt und in der Wissensbank durch Formeln oder Konstanten ersetzt werden. Durch die Zwischenschaltung einer wissensbasierten Interpretation der Genotypen entsteht somit eine sehr flexible und mächtige Optimierungsplattform.

Wissensbasierte Operationen erfolgen strukturiert im Kontext eines zugehörigen Problemraums. Die Elemente des Problemraums bestehen im ersten Schritt aus einer leeren „Hülle“ eines Tragwerkmodells. Das Faktenwissen, in Form der im vorigen Kapitel definierten Beschreibungssprache, kommt zunächst zur Erzeugung des Topologiemodells zum Einsatz. Hierzu werden Informationen wie Grundrisstyp und numerische Designparameter aus dem Faktenwissen extrahiert und zur wissensba-

sierten Weiterverarbeitung herangezogen. Teilweise können hier auch direkt Parameter ohne wissensbasierte Verarbeitung weiterverarbeitet werden. Durch Erstellung des Topologiemodells wurde der Problemraum um eine Gruppe strukturierender Elemente erweitert. In einem zweiten Schritt erfolgt die schrittweise Anwendung der Design-Selektionen auf sämtliche Elemente des Topologiemodells. Das vereinfachte Modell unterscheidet hierbei zunächst die Kanten und die Knoten des Topologiemodells. Die Testumgebung verwendet Design-Selektionen ausschließlich zur Anwendung auf Kanten, was jedoch systemtechnisch nicht auf diesen Elementtyp begrenzt ist.

Jede Design-Selektion beinhaltet zunächst einen Satz von Anwendungsbedingungen. Diese Bedingungen sind flexibel auf die vorherrschenden Verhältnisse einer Topologiekante anpassbar. Die wichtigste Bedingung ist die geometrische Einordnung der Kante. An einer horizontalen Kante ist es nicht sinnvoll, die Erstellung von Stützen zu erlauben, dies trifft nur auf Vertikale Kanten zu. Weiterhin ist zu untersuchen, ob in der Umgebung der Kante aussteifende Elemente (Kerne, Outrigger, etc.) vorgesehen sind. Hierdurch kann möglicherweise die Entscheidung zur Erstellung einer Wand getroffen werden, wodurch die umgebenden vertikalen Kanten im betrachteten Beispiel keine Stützen ausbilden. Ebenso werden beispielsweise inmitten eines Kernes keine Deckenelemente erstellt, um eine Öffnung zur vertikalen Erschließung vorzusehen. Die derart formalisierten Bedingungen sind auf eine unzählige Reihe möglicher Konstellationen erweiterbar. Sie können sich sowohl auf wissensbasiert ermittelbare Fakten als auch auf Fakten beziehen, die durch den Anwender in der Bildungsvorschrift ergänzt wurden und somit Teil der Optimierung sind.

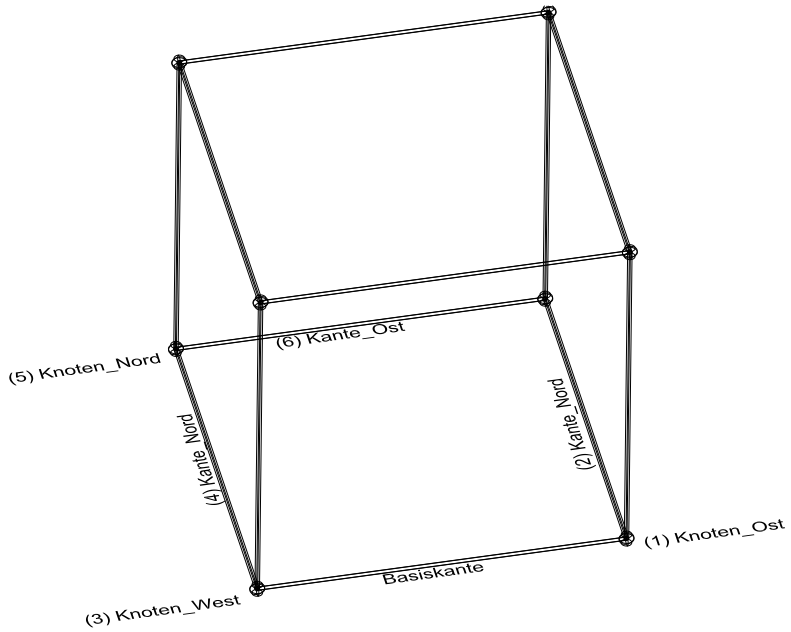


Abbildung 5-4: Bestimmung benachbarter Kanten innerhalb des Topologiemodells

Weitere Parameter, die durch Inferenz von Elementen der Wissensbasis ermittelt werden, sind die Position und Geometrie der zu erzeugenden Elemente sowie die zugehörigen Topologiekanten. Hierbei erfolgt der Zugriff topologischer Kanten untereinander durch geometrische Iteration zwischen Kanten und Knoten. Im Fall des in dieser Arbeit angewandten vereinfachten Modells kann die Beschreibung der geometrischen Lage der Elemente zueinander anhand von rechtwinkligen Richtungsan-

gaben erfolgen. Bei Übertragung auf komplexere Geometrien muss hier eine besser geeignete Geometriebeschreibungsform gewählt werden [BOR07]. Abbildung 5-4 zeigt beispielhaft die Bestimmung benachbarter Topologieelemente zur Erstellung eines Deckenelementes, ausgehend von einer Kante, die hier als „Basiskante“ bezeichnet wird. Die Kanten (2) und (4) sind jeweils mit "Kante Nord" bezeichnet. Diese zunächst verwirrende Bezeichnung ist relativ zu den umgebenden Knoten zu sehen, aus deren Kontext die Kanten jeweils angesprochen werden. So ist Kante (4) aus Sicht des Knoten (5) als "Kante Süd" anzusprechen. Eine detaillierte Auflistung der Berechnungselemente der Design-Wissensbank ist dem Anhang zu entnehmen.

5.4 Finite Elemente Berechnung

In [PUL03] wurden bereits erste Experimente zur Optimierung von Stahlbeton-Tragwerken auf der Basis eingeschossiger Modellierung durchgeführt. Die in diesem Forschungsprojekt angewandte Näherungslösung basierte auf einem äquivalenten Balkenmodell zur Repräsentation von Wänden. Die Deckenelemente wurden zur Bestimmung der Eigenfrequenzen vereinfachend als Ersatzmassen abgebildet. Die Verwendung des vereinfachten Balkenmodells hat sich in [KAR05] als unzureichend herausgestellt. Insbesondere im Fall der zu erwartenden unsymmetrischen Tragwerke, die während des Optimierungsprozesses untersucht werden, weisen Vergleichsrechnungen zwischen dem vereinfachten Balkenmodell und genaueren Finite Element (FE-)-Berechnungen erhebliche Abweichungen auf. Basierend auf diesen Erkenntnissen wurde im Rahmen der vorliegenden Arbeit auf die Anwendung einer Näherungslösung verzichtet und die Ermittlung der maßgeblichen Schnittgrößen anhand vollständiger FE-Berechnungen durchgeführt.

Zur Bestimmung der Schnittgrößen mittels des externen FE-Systems B&B [THI04] wird das Strukturmodell in ein dreidimensionales Finite-Elemente Netz überführt. Das Tragwerk wird hierbei als ein monolithischer Verbund der einzelnen Strukturelemente idealisiert. Für die unteren Tragglieder wird eine eingespannte Lagerung angenommen.

Die FE-Modellierung des Strukturmodells beinhaltet die Abbildung der Strukturelemente entsprechend ihres Typs durch eine geeignete Anzahl an finiten Elementen. Die Modellierung von Stützen und Unterzügen erfolgt mittels Balkenelementen, die in B&B dem Typ 111 entsprechen [THI04]. Decken und Wände werden durch Schalenelemente des B&B Typ 215 repräsentiert.

Die Erstellung des FE-Modells erfolgt unter Anwendung der wissensbasierten Inferenzmechanismen. Hierbei wird sowohl auf Berechnungselemente der hinterlegten Wissensbanken der Fachdisziplin "Tragwerksplanung", als auch auf Parameter der genotypischen Beschreibung des betreffenden Individuums zurückgegriffen. Auf diese Art werden beispielsweise die für die FE-Berechnung erforderlichen Querschnittswerte und Materialkennwerte einzelner Strukturelemente sowie die Lastannahmen ermittelt. Zur Lastermittlung für die FE-Berechnung wurden folgende Lastannahmen getroffen:

- **Ständige Lasten:** Eigengewicht der Tragstruktur sowie Ausbaulast gem. DIN 1055-3
- **Verkehrslasten:** Nach DIN 1055-3 ist für Büroräume eine verteilte Flächenlast von q_k =2,0 kN/m² vorgesehen, die durch einen Aufschlag für leichte Trennwände um q_k'=0,75 kN/m² zu einer verteilten Flächenlast von 2,75 kN/m²

angenommen wird. Nach [KOE03] wird in der gängigen Baupraxis jedoch die Bemessung der Decken unter Annahme einer Last von q_k=5,0 kN/m² durchgeführt. Zur Lastweiterleitung und zur Bemessung der vertikalen Tragstruktur wird eine abgeminderte Verkehrslast von q_k=3,5 kN/m² angenommen. In dem vorliegenden Anwendungsbeispiel wird vereinfachend die abgeminderte Verkehrslast von q_k=3,5 kN/m² zur Bemessung der gesamten Tragstruktur verwendet.

- **Windlasten:** Die dynamische Windwirkung wird entsprechend DIN 1055-4 [HOL07] ermittelt. Da für Hochhäuser eine Untersuchung der dynamischen Windwirkung erforderlich ist, erfolgt zunächst eine dynamische Berechnung, die zur Ermittlung des Böenreaktionsfaktors *G* führt. Aus den einwirkenden statischen Windlasten sowie dem Böenreaktionsfaktor *G* lassen sich quasi-statische Ersatzlasten zur Bestimmung der effektiv auf das Bauwerk wirkenden Windkräfte ermitteln.
- **Erdbebenlasten:** Zur Ermittlung der Lasten aus Erdbebenwirkung kommt das Antwortspektrenverfahren entsprechend DIN 4149 (04.05) [HOL07] zum Einsatz. Die dynamische Untersuchung findet anhand des Bemessungsspektrums getrennt für Bodenbeschleunigungen parallel zur x- und y- Koordinate statt. Die sich hieraus ergebenden quasi-statischen Ersatzschnittkräfte sind Modallasten, die je Eigenform des Tragwerks einem eigenen Lastfall entsprechen. Die Summe der zu berücksichtigenden Modallasten muss mindestens 90% der Gesamtmasse des Bauwerks ausmachen, woraus sich die zu berücksichtigende Anzahl der Eigenformen ergibt. Die Kombination der Lastfälle aus Modallasten erfolgt mittels der vollständigen quadratischen Kombination (CQC-Regel).

Die Berechnung des FE-Modells erfordert ein zweistufiges Vorgehen. Im ersten Schritt erfolgt die dynamische Analyse zur Ermittlung von Eigenformen, Eigenfrequenzen sowie Ersatzlasten für Einwirkungen aus Wind- und Erdbebenlasten. Systembedingt muss die dynamische Analyse getrennt für X- und Y-Richtung durchgeführt werden. Die Ergebnisse der dynamischen Analyse sowie die statischen Lastfälle bilden die Ausgangsbasis für die im nächsten Schritt durchzuführende statische Analyse.

5.5 Wissensbasierte Bemessung

Die Bewertung eines Bauwerks umfasst neben der Schnittgrößenermittlung die wissensbasierte Bemessung und darauf basierend die überschlägliche Ermittlung der Baukosten sowie weiterer Parameter. Ziel der wissensbasierten Bemessung ist die Bewertung der Tragfähigkeit sowie die vereinfachte Bemessung einzelner Bauteile der Tragstruktur, die als Grundlage für die Kostenermittlung dienen. Die Bemessung erfolgt anhand der maßgebenden Schnittgrößen, die durch die externe FE-Komponente B&B ermittelt wurden.

Beispielhaft soll hier die vereinfachte Bemessung eines einzelnen Strukturelementes einer Deckenplatte auf Biegung unter Anwendung der Bemessungs-Wissensbank erläutert werden. Zur Veranschaulichung erfolgt zunächst eine manuelle Bemessung auf reine Biegung nach DIN 1045-1 mittels dimensionsloser Beiwerte (ω-Tabelle).

In dem hier vorgestellten vereinfachten Beispiel besteht die gesamte Deckenplatte eines Geschosses aus einem einzigen Strukturelement. In Modellen mit mehreren

Grundrissachsen wird jeder von Achsen umschlossene Flächenteil als einzelnes Strukturelement einer Deckenplatte angesehen. Somit würde die hier beispielhaft vorgestellte Bemessung einer Deckenplatte bei einem größeren Bauwerk entsprechend für jedes Deckenfeld zwischen den Gebäudeachsen individuell erfolgen. Ebenso durchlaufen sämtliche weiteren Strukturelemente (Stützen, Wände, Balken) eine eigene Bemessung, da aufgrund der FE-Berechnung je Strukturelement unterschiedliche Schnittgrößen ermittelt werden können.

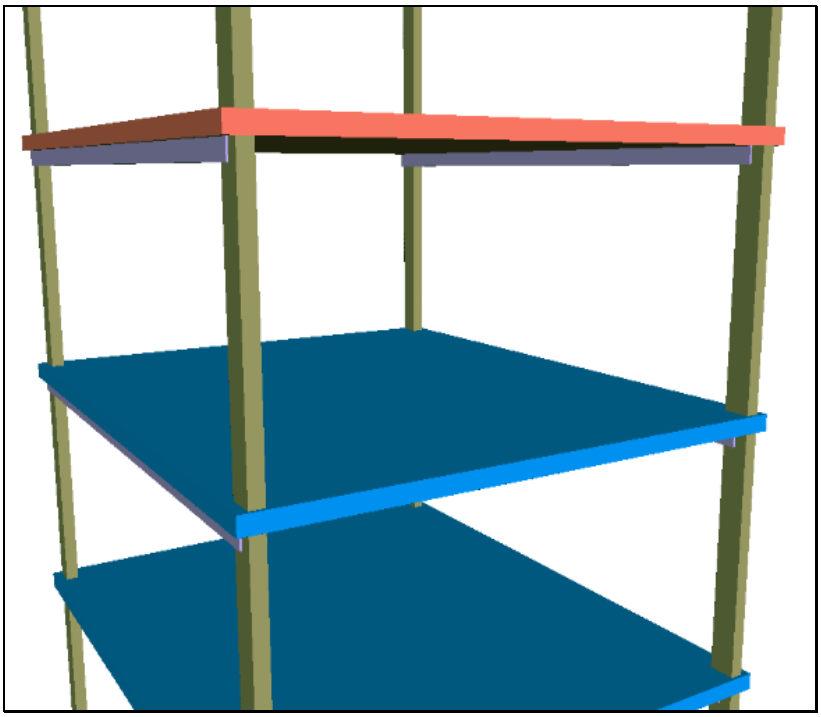


Abbildung 5-5: Systemdarstellung des zu bemessenden Deckenelementes

Manuelle Bemessung nach DIN 1045-1 (2008-08):

Abmessungen der Deckenplatte:

Breite (Achsmaß): $dx = 6{,}00\,m$

Länge (Achsmaß): $dy = 7{,}20\,m$

Deckenstärke: $h = 0{,}20\,m$

Streifenbreite: $b = 1{,}00\,m$

Umweltklasse XC1: $c_{nom} = 20\,mm$

Durchmesser der Biegezugbewehrung: $d_{sl} = 10\,mm$

Statische Höhe:

$$d = h - \frac{c_{nom}}{1000} - \frac{d_{sl}}{2 \cdot 1000} = 0{,}2 - \frac{20}{1000} - \frac{10}{2 \cdot 1000} = 0{,}175\,m \qquad (5\text{-}1)$$

Schnittgrößen aus FE-Berechnung:

$M_{Ed} = 47{,}37\ kNm \qquad N_{Ed} = 0\,kN \qquad z_S = 0\,m$

Bemessungsmoment der Deckenplatte :

$$M_{Eds} = M_{Ed} - N_{Ed} * z_s = 47{,}37 - 0 \cdot 0 = 47{,}37\ kNm \tag{5-2}$$

Betonfestigkeitsklasse C30/37:

$f_{ck} = 30\,N/mm^2 \qquad \gamma_c = 1{,}5$

Abminderungsbeiwert zur Berücksichtigung von Langzeiteinwirkungen auf die Druckfestigkeit für Normalbeton:

$\alpha = 0{,}85$

Bemessungs-Druckfestigkeit:

$$f_{cd} = \alpha \cdot \frac{f_{ck}}{\gamma_c} = 0{,}85 \cdot \frac{30}{1{,}5} = 17{,}0\,N/mm^2 \tag{5-3}$$

Dimensionsloses Moment:

$$\mu_{Eds} = \frac{M_{Eds} \cdot 0{,}001}{b \cdot d^2 \cdot f_{cd}} = \frac{47{,}37 \cdot 0{,}001}{1{,}00 \cdot 0{,}175^2 \cdot 17} = 0{,}0910 \tag{5-4}$$

Aus ω–Tabelle[5] abgelesen (interpoliert):

$\omega = 0{,}1010$

Ermittlung der Zugbewehrung:

$$\sigma_{sd} = \frac{500 N/mm^2}{1{,}15} = 435{,}00\,N/mm^2$$

$$A_{s,erf} = \frac{\omega \cdot b \cdot d \cdot f_{cd} + N_{Sd}}{\sigma_{sd}} \cdot 10000 = \frac{0{,}1010 \cdot 1 \cdot 0{,}175 \cdot 17{,}0 + 0}{435{,}000} \cdot 10000 = \underline{6{,}91 cm^2/m} \tag{5-5}$$

[5] Tabelle 4.2 für C12/15-C50/60, ohne Druckbewehrung, [BET02], Seite 300

Wissensbasierte Bemessung:

Die wissensbasierte Bemessung erfolgt im Rahmen der Fitness-Evaluation für jedes einzelne Strukturbauteil. Das auslösende Ereignis der Bemessung erfolgt indirekt durch Inferenz der Teilergebnisse im Zuge der Ermittlung eines globalen Fitnesswertes. Im Verlauf weiterer Berechnungen ist die Bemessung der Tragfähigkeit und des Bewehrungsgehalts jedes Bauteils erforderlich, was im Fall der betrachteten Decke die Ermittlung der Biegezugbewehrung $A_{s,erf}$ erfordert. Dieser Wert ist der Ausgangspunkt und das Endergebnis der in Tabelle 5-1 dargestellten deklarativen Bemessung.

Tabelle 5-1: Wissensbasierte Bemessung einer Deckenplatte - Erklärungskomponente

Slab S_EX_F004 (12.80m)/X1/Y1 (145)

Parameter	*Wert*	*Einh.*	*Ermittlung*
Betonfestigkeitsklasse	C30/37		PM-Zugriff
dx	6.4000	m	PM-Zugriff
dy	7.6000	m	PM-Zugriff
mue_Eds_rechnerisch	0.0910		Formel
mue_Eds_lim	0.2960		Konstante
M_Eds	47.3691	kNm	Formel
b	1.00	m	Konstante
d	17.5000	cm	Formel
f_cd	17.0000	N/mm²	Tabelle
M_ed	47.3691	kNm	PM-Zugriff
N_ed	0.0000	kN	PM-Zugriff
z_s	0.0000	m	PM-Zugriff
h	0.2000	m	PM-Zugriff
c_nom	20.0000	mm	Formel
d_sl_1	10.0000	mm	Konstante
alpha_C	0.8500		Konstante
f_ck	30.0000	N/mm²	Tabelle
c_min	10.0000	mm	Tabelle
delta_c	10.0000	mm	Tabelle
Expositionsklasse	XC1	-	PM-Zugriff
sigma_s1_d	435.0000	N/mm²	Formel
omega_1	0.0957	-	Tabelle
mue_Eds	0.0910	-	Formel
erf_A_sl	**6.5450**	**cm²/m**	**Formel**

5.6 Fitness-Evaluation

5.6.1 Optimierungskriterien

Die Bewertung des Tragwerkes kann anhand unterschiedlichster Kriterien erfolgen, die durch den Anwender festzulegen sind. Im vorliegenden Anwendungsbeispiel besteht primär das Ziel der Optimierung wirtschaftlicher Kriterien. Die Bewertung der Wirtschaftlichkeit eines Tragwerks geht weit über die klassischen Optimierungsansätze, die oftmals eine reine Gewichtsminimierung anstreben, hinaus. Realitätsnahe Bewertungskriterien, die an ein zu optimierendes Bauwerk gestellt werden, können sehr viele Randbedingungen umfassen, die je nach Planungssituation variieren und relativ häufig angepasst werden müssen.

Zunächst muss festgelegt werden, welche Ziele mit der Optimierung verfolgt werden sollen. Diese sind abhängig von der Rolle des Planers sowie der gesamten Planungs- und Bausituation. Ist der Planer daran interessiert, die reinen Baukosten zu minimieren, ergeben sich völlig andere Kriterien als im Fall ausschließlicher oder zusätzlicher Beachtung des ökonomischen Betriebs eines Bauwerks während der gesamten Nutzungsphase. Im Folgenden sind einige mögliche Optimierungskriterien zusammengestellt:

Baukosten:

- Planungskosten (Komplexität der Tragwerks)
- Materialkosten
- Lohnkosten
- Erhöhte Kosten für Ausbau, beispielsweise komplexe Leitungsführung aufgrund von Unterzügen
- Bauzeit, bedingt durch mögliche vertragliche Bindungen, Kosten der Vorfinanzierung
- etc.

Nutzungsphase:

- Nutzfläche
- Mietertrag
- Repräsentative Qualität
- Variabilität der Grundrissgestaltung
- Energiebedarf
- Tageslicht-Situation
- Kosten für Rückbau
- etc.

Die genannten Punkte stellen lediglich einen ersten Anhaltspunkt für mögliche Optimierungskriterien dar. Abhängig von der Planungssituation und Zielsetzung können

einige dieser Kriterien als signifikant angesehen werden. Daher ist ein möglichst flexibles Verfahren der softwaretechnischen Beschreibung der Optimierungskriterien erforderlich. Der im Rahmen der vorliegenden Arbeit entwickelte Ansatz ermöglicht die Verwendung und Kombination beliebiger weiterer Kriterien, sofern sie sich aus den vorhandenen Modelldaten mit Hilfe des wissensbasierten Ansatzes bestimmen lassen.

5.6.2 Bewertungsindex

Der Bewertungsindex (5-9) ermittelt sich aus einer Kombination sämtlicher zu optimierender Kriterien. Das entwickelte System ermöglicht die Betrachtung einer enormen Bandbreite an Optimierungskriterien. Das vorgestellte Anwendungsbeispiel soll die prinzipielle Vorgehensweise verdeutlichen, daher wurde der Schwerpunkt bewusst auf die folgenden Kriterien gelegt:

- Index der Rohbaukosten (Aufwandsindex)
- Index des Mietertrages (Ertragsindex)

Das vorliegende Anwendungsbeispiel umfasst hierbei ausschließlich die Betrachtung der reinen Stahlbeton-Tragstruktur. Für die Optimierung ist es nicht erforderlich, tatsächlich korrekte Baukosten und Mieterträge zu erfassen. Die betrachteten Modelle weisen hierfür bei weitem nicht den erforderlichen Detaillierungsgrad auf. Ein höherer Detaillierungsgrad ist während der Optimierungsphase auch nicht notwendig, da die Modelle lediglich die Vergleichbarkeit unterschiedlicher Entwürfe ermöglichen sollen. Eine Bewertungsfunktion muss daher primär einen hinreichend genauen Vergleich unterschiedlicher Gebäudevarianten in Bezug auf die gewünschten Parameter ermöglichen.

Der Index der Rohbaukosten (5-7) setzt sich zusammen aus den Materialkosten und den fiktiven Lohnkosten. Die Materialkosten ermitteln sich aus dem Betonvolumen, sowie der erforderlichen Menge an Betonstahl. Die Lohnkosten werden in einem linearen Zusammenhang zu den Materialmassen angenommen und können somit über entsprechende Faktoren ermittelt werden. Es kann vereinfachend davon ausgegangen werden, dass die tatsächlich zu erwartenden Herstellungskosten eines Bauwerks sowie weitere Kosten der Baunutzungsphase in einem stetigen Verhältnis zu dem ermittelten Kostenindex stehen, so dass eine ausreichende Vergleichbarkeit des Aufwandes gegeben ist.

Der Index des Mietertrags (5-8) ermittelt sich aus der vermietbaren Nutzfläche sowie einer anzusetzenden Flächenmiete über einen bestimmten Nutzungszeitraum. Die Nutzfläche wird vereinfachend aus der Summe der Deckenflächen abzüglich der Summe der Konstruktionsflächen aus Stützen und Wänden sowie Durchbrüchen für vertikale Erschließung ermittelt. Auch für diesen Index kommen lediglich stark vereinfachte Annahmen zum Einsatz, die nicht zur Ermittlung tatsächlicher Mieterträge ausreichen, jedoch eine angemessene Vergleichbarkeit der Erträge ermöglichen.

Im Fall einkriterieller Optimierung ist es erforderlich, die gewählten Kriterien zu einem übergeordneten Optimierungskriterium zusammenzufassen. Im Fall des vorliegenden Beispiels wird als übergeordnetes Optimierungskriterium das Verhältnis aus Aufwands- und Ertragsindex herangezogen. Es erfolgt keine Berücksichtigung finanzmathematischer Aspekte wie z. B. Finanzierungskosten. Ziel der Optimierung ist die

Minimierung des Bewertungsindex. Ebenso wäre eine multikriterielle Optimierung möglich, die beide Indizes (oder ggf. weitere) als unabhängige Optimierungsparameter heranzieht.

$$K_m = \sum_{Sw=1}^{n_{Sw}} \sum_{Se=1}^{n_{Se}} V_{b,Se} \cdot I_b + G_{s.Se} \cdot I_s \tag{5-6}$$

$$K_g = K_m \cdot I_g \tag{5-7}$$

$$E = I_e \cdot \sum_{Sw=1}^{n_{Sw}} \sum_{Se=1}^{n_{Se}} A_d - A_{kv} \tag{5-8}$$

$$F = \frac{K_g \cdot \left(1 + \sum_{Sw=1}^{n_{Sw}} \sum_{Se=1}^{n_{Se}} Sf_{Se} \right)}{E} \tag{5-9}$$

I_b : Kostenindex Beton

I_s : Kostenindex Stahl

I_g : Kostenindex Gesamtbaukosten bezogen auf Materialeinsatz

I_e : Kostenindex Ertrag bezogen auf Nutzfläche

Se : Strukturelement

Sw : Stockwerk

$V_{b,\,Se}$: Materialvolumen Beton je Strukturelement

$G_{s,\,Se}$: Materialgewicht Stahl je Strukturelement

Sf_{Se} : Summe Straffaktoren je Strukturelement

K_m : Materialkosten

K_g : Baukosten gesamt

A_d : Horizontale Fläche der Deckenelemente

A_{kv} : Konstruktionsfläche vertikaler Strukturelemente

E : Ertrag gesamt (Index des fiktiven monatlichen Mietertrags)

F : Bewertungsindex (=Fitness des Gesamtsystems)

Für den Fall, dass unter den maßgebenden Lastfallkombinationen ein oder mehrere Strukturelemente geringfügige Überschreitungen der maximal aufnehmbaren Schnittgrößen aufweisen, kann der Bewertungsindex mit entsprechenden Straffaktoren beaufschlagt werden. Hierdurch wird erreicht, dass Tragwerke, die trotz leichter Schwächen der Struktur insgesamt gute Lösungen darstellen, als Grundlage weiterer genetischer Operationen Verwendung finden können.

5.7 Visualisierung

Das integrierte Entwurfssystem beinhaltet eine Komponente zur dreidimensionalen Visualisierung der entwickelten Tragstrukturen. Der Anwender kann sich mit Maus und Tastatur frei um das Tragwerk herum bewegen und auch in dieses „hineingehen“. Eine optimierte Maussteuerung ermöglicht komfortable Steuereigenschaften und intuitive Bewegungsabläufe. Zusätzlich kann der Anwender direkt in der Visuali-

sierung einzelne Elemente „anklicken“, um detaillierte Informationen mit Hilfe der integrierten Erklärungskomponente zu erhalten.

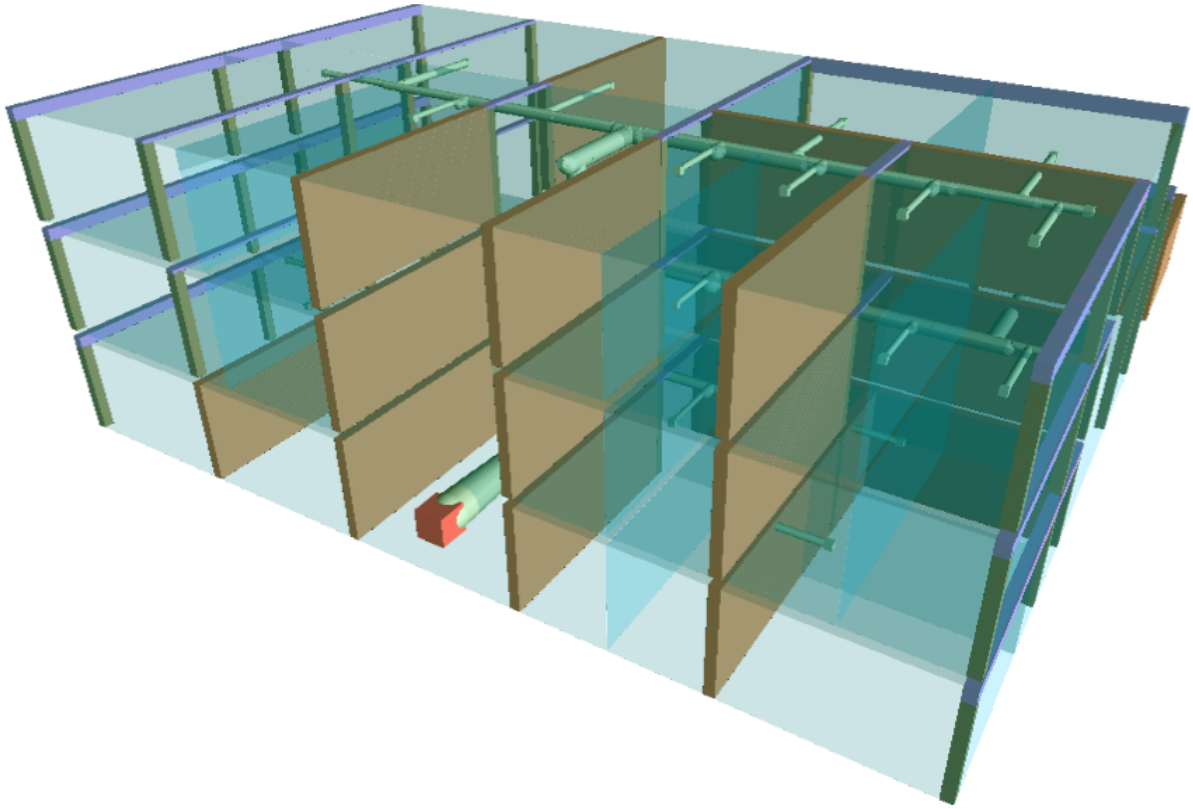


Abbildung 5-6: Visualisierung mittels Direct3D

Die Visualisierungskomponente basiert auf der Technologie Direct3D, eine Komponente der DirectX-Technologie der Firma Microsoft. Hierbei handelt es sich um eine Programmierschnittstelle zur Darstellung virtueller Welten. Diese wird häufig für Computerspiele verwendet, bietet jedoch leistungsfähige Darstellungsfunktionen, die einen Einsatz für beliebige dreidimensionale Visualisierungsanwendungen ermöglicht. Die in Abbildung 5-6 dargestellte Visualisierung stellt beispielhaft ein manuell entworfenes Tragwerksmodell dar. Zusätzlich zu den in der vorliegenden Arbeit beschriebenen Tragwerkselementen sind in diesem Beispiel Elemente der Raumlufttechnik dargestellt. Gleichzeitig wurden in dem vorliegenden Beispiel die Deckenelemente ausgeblendet und begrenzende Raumvolumina transparent dargestellt.

Die hier dargestellten Elemente stellen einen Ausblick dar, der zeigt, wie die Konzepte der vorliegenden Arbeit im Zusammenspiel unterschiedlicher Gewerke interdisziplinär zur Anwendung kommen können. So kann, wie in der Visualisierung dargestellt, das Gewerk der Raumlufttechnik in den Planungsprozess mit eingebunden werden. Die Bewertung des Gebäudeentwurfes beinhaltet in diesem Fall neben den bereits dargestellten Aspekten der Tragwerksplanung zusätzliche Kriterien, die sich aus den Belangen der Raumlufttechnik sowie aus der Interaktion der Gewerke Tragwerksplanung und Raumlufttechnik ergeben. So kann beispielsweise eine gefundene Aussteifungslösung eigenständig betrachtet zunächst als sehr wirtschaftlich erscheinen. Unter Berücksichtigung der Raumlufttechnik und den damit notwendigen Installationsöffnungen führt die gesamtheitliche Betrachtungsweise unter Umständen zu völlig anderen Bewertungen eines Entwurfs.

5.8 Verteiltes Rechnen

Trotz umfangreicher Analyse nach Möglichkeiten der Komplexitätsreduktion ist die Simulation und Fitness-Berechnung eines vollständigen und komplexen Tragwerksmodells mit erheblichen Rechenzeiten verbunden.

Die grundlegende Idee der hier verwendeten Architektur des verteilten Rechnens basiert auf der *Berkeley Open Infrastructure for Network Computing (BOINC)* [BOI09]. Erste Ansätze der breiten Aktivierung ungenutzter Rechenleistung gehen auf das Projekt *SETI@home (Search for ExtraTerrestrial Intelligence)* der UC Berkley (Kalifornien, USA) zurück, das inzwischen auf der BOINC-Plattform weitergeführt wird. Mittels des weltweit empfindlichsten Radioteleskops werden Funksignale aus dem Weltraum empfangen und auf außerirdische Inhalte hin untersucht. Die dabei anfallenden Datenmengen sind auch mit Großrechnern kaum zu bewältigen. Die Daten werden daher in Arbeitspakete zerlegt und durch einen Server über das Internet auf die Rechner von freiwilligen Teilnehmern weitergeleitet. Dort erfolgt die zeitaufwändige Datenanalyse und anschließende Rücksendung der Ergebnisse. Inzwischen sind ca. 4.000.000 Teilnehmer registriert, die in Zeiten geringer Auslastung die Rechenleistung ihrer Computer zur Verfügung stellen.

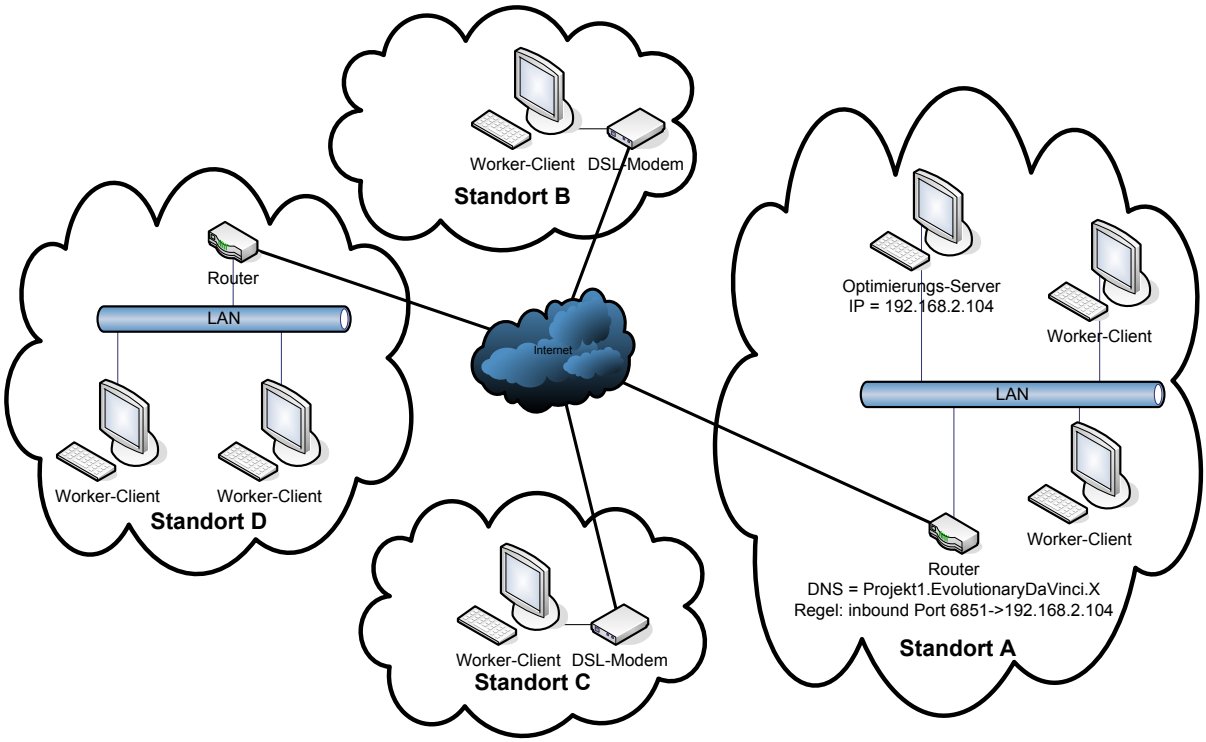


Abbildung 5-7: Netzwerk-Topologie für verteiltes Rechnen

Zur Minimierung der Gesamtrechenzeit wurden in der vorliegenden Arbeit verschiedene Möglichkeiten der Parallelisierbarkeit der Prozesse auf mehreren Computersystemen untersucht. Ein maßgeblicher Teil der Rechenzeit wird für die dynamische Untersuchung des Tragverhaltens aufgewendet.

Da diese jeweils in X- und Y-Richtung getrennt voneinander erfolgt, besteht hier direkt die Möglichkeit der Aufteilung dieser Prozesse auf genau zwei Rechner. Die weiteren Schritte der Fitness-Ermittlung wären in sich weitgehend parallelisierbar, müssten jedoch sequentiell abgearbeitet werden, weshalb der Rechenaufwand für Verteilung und Kommunikation erheblich wäre.

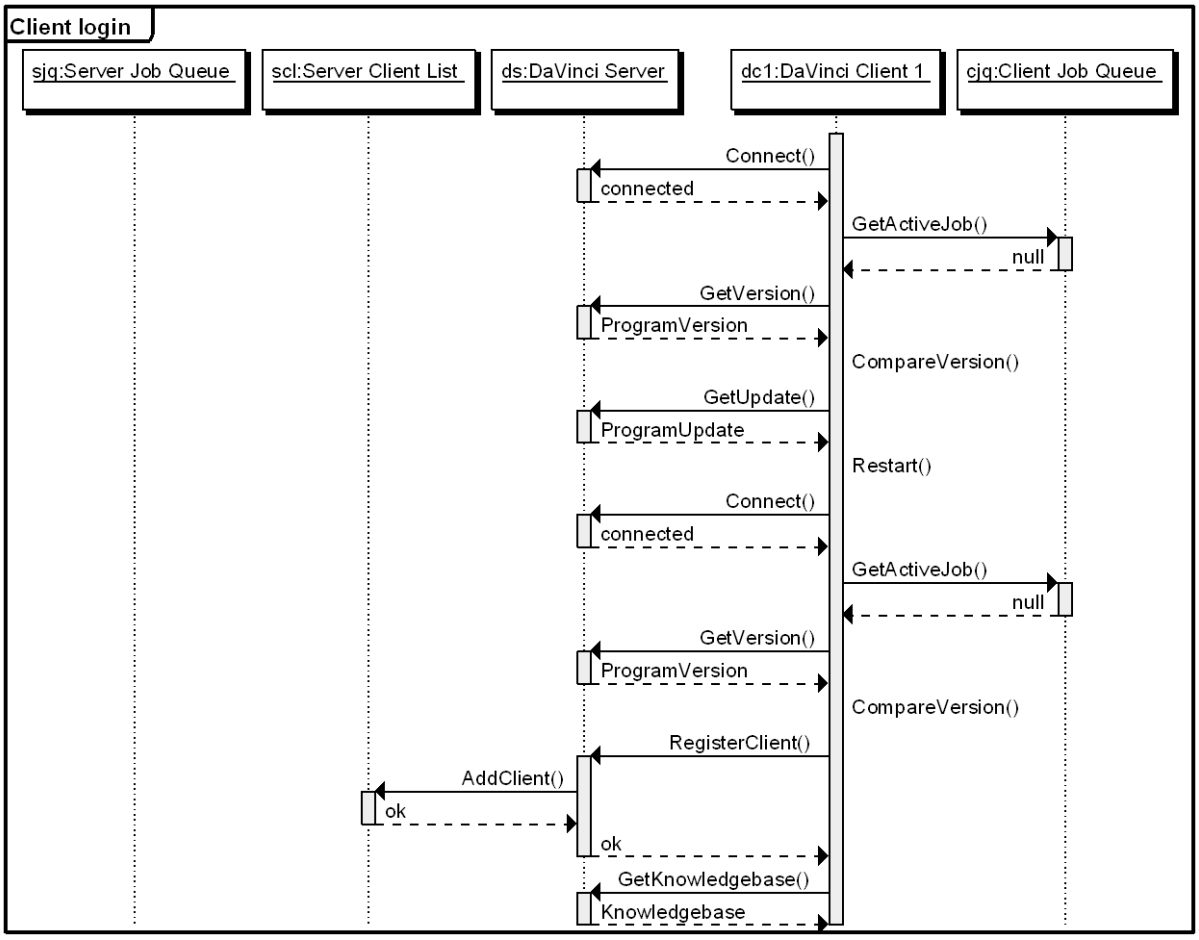


Abbildung 5-8: Sequenzdiagramm des Kommunikationsaufbaus

Das größte Potential verteilter Berechnung besteht in der Delegation vollständiger Individuen einer geschlossenen Generation an die Clients, mit dem Ziel der Berechnung der Fitnesswerte. Bedingt durch den geringen Kommunikationsaufwand ist somit eine Paketierung und Verteilung der Arbeitsdaten an beliebige Clients über das Internet problemlos möglich (Abbildung 5-9). Hierbei werden folgende Vorteile erzielt:

- **minimaler Verteilungsaufwand:** Die Verteilung bzw. Separation der Arbeitspakete erfolgt bereits durch das Optimierungsverfahren in Form der einzelnen Individuen.
- **minimaler Kommunikationsaufwand:** Die Worker-Clients erhalten zu Beginn eines Optimierungsvorganges allgemeingültige Daten (Wissensbanken, BNF-Definition, etc.). Für die einzelnen Individuen sind dann je Berechnungsschritt lediglich deren String-Repräsentation und die ermittelten Fitnesswerte auszutauschen.
- **Robuster Betrieb:** Durch lange unabhängige Laufzeit jedes Clients kann die Kommunikation während der Bearbeitung für längere Zeit unterbrochen werden, ohne andere Clients zu blockieren.
- **Dynamische Client-Kopplung:** Die Clients können im laufenden Betrieb dynamisch ergänzt und getrennt werden. Lediglich bis zum Zeitpunkt der Tren-

nung berechnete Werte eines Individuums können bei Entfernen eines Clients verloren gehen und müssen später von einem anderen Client nochmals berechnet werden.

- **Automatisierte Software-Updates:** Zu Beginn eines jeden Optimierungsvorgangs findet eine Prüfung der Programmversion auf den Worker-Clients statt, ggf. erfolgt ein automatisiertes, unbeaufsichtigtes Update und der Neustart der betroffenen Clients. Der Ablauf einer Client-Authentifizierung und anschließendem automatisiertem Client-Update ist in Abbildung 5-8 dargestellt.
- **Minimaler Konfigurationsaufwand:** Es sind keinerlei client-seitigen Netzwerkeinstellungen erforderlich, nur der DNS-Eintrag und Login des Optimierungsservers müssen einmalig definiert werden (siehe Abbildung 5-7).
- **Sicherheit:** Ein mehrstufiges Sicherheitssystem stellt die Authentizität und Autorisierung der Clients sicher. Hierbei wird die Integrität der zugrunde liegenden Wissensbanken, der zu bewertenden Individuen sowie sämtlicher Komponenten der Client-Software durch mehrstufigen Austausch von Prüfsummen sichergestellt. Weiterhin besteht die Möglichkeit, die Kommunikation mittels einer verschlüsselten Verbindung über einen VPN-Tunnel abzusichern, beispielsweise per OpenVPN [ZEL08].

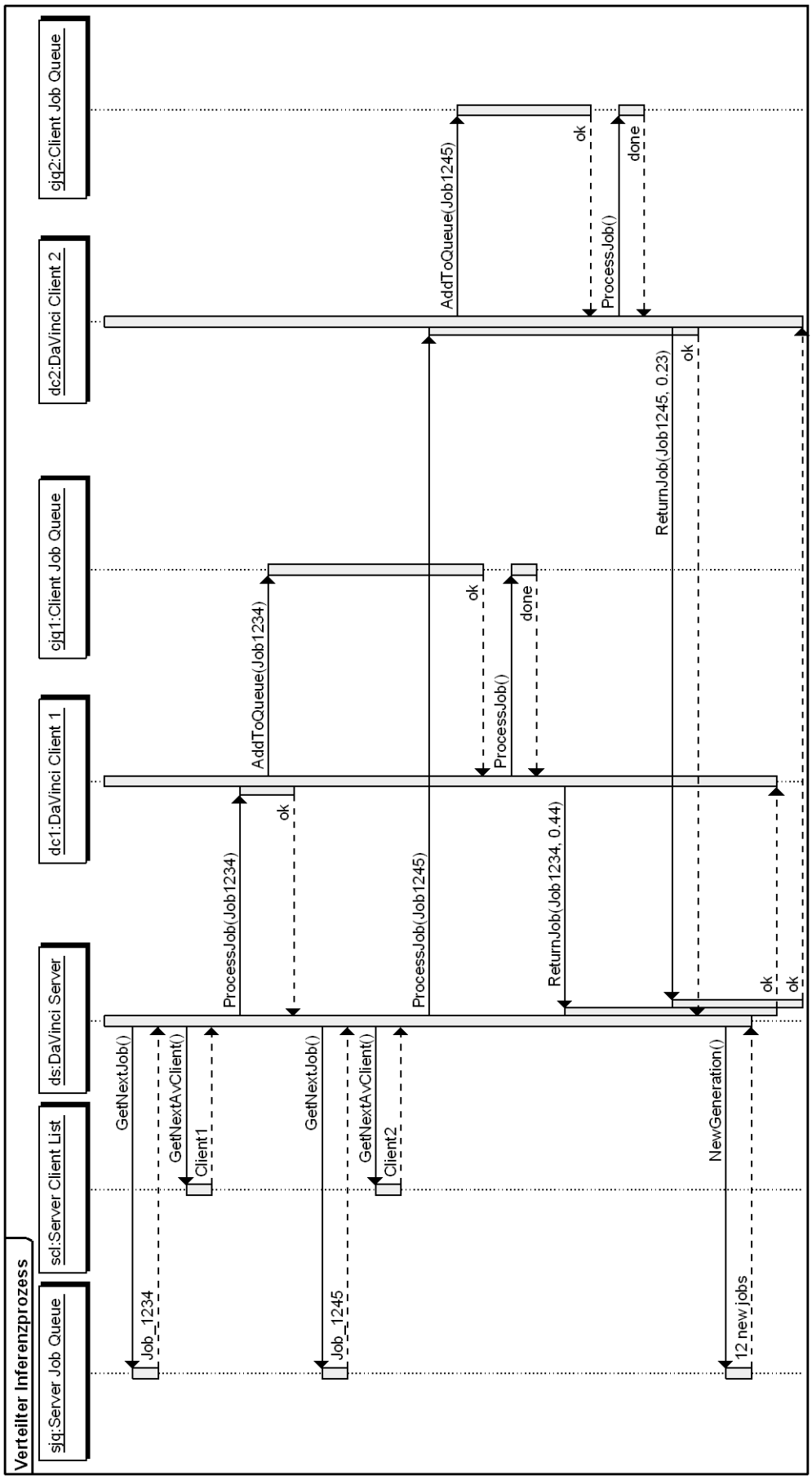


Abbildung 5-9: Sequenzdiagramm des Verteilten Optimierungs- und Inferenzprozesses

5.9 Prototypische Implementierung

Die in der vorliegenden Arbeit entwickelten Konzepte wurden prototypisch in mehreren Softwareapplikationen implementiert. Die gesamte Implementierung umfasst folgende Komponenten:

- Wissensakquisitionskomponente KBDT
- GP Definitionskomponente GPCore-Manager
- Evolutionary DaVinci, Client und Server, bestehend aus grafischer Design- und Erklärungskomponente, Laufzeitumgebung für die genetische Optimierung, Inferenzkomponente, Schnittstelle zum FE-Programmsystem B&B, Netzwerk-Infrastruktur zur Unterstützung verteilten Rechnens (Abbildung 5-10)

Laufzeit-Umgebung

Die Laufzeit-Umgebung für den Optimierungs-Server sowie auch für die Worker-Clients erfordert Microsoft Windows 2000, Windows XP, Windows Vista, Windows 7 oder ein dazu kompatibles Betriebssystem. 2-4 GB Arbeitsspeicher sind empfehlenswert. Da die Clients keinerlei Visualisierung durchführen, sind an deren Grafikleistungen keine besonderen Anforderungen zu stellen. Der Server hingegen sollte eine hardwareseitige DirectX-Grafikbeschleunigung unterstützen.

Programmierumgebung

Für die Implementierung sämtlicher Komponenten wurde die objektorientierte Programmiersprache C++ in Verbindung mit den Microsoft Foundation Classes (MFC) verwendet [LEI98]. Für die grafische Darstellung der Optimierungsergebnisse kam Microsoft DirectX Version 9 zum Einsatz.

Netzwerk-Funktionalitäten

Der Optimierungs-Server muss netzwerkseitig in der Lage sein, Verbindungen an Port 6851 TCP anzunehmen. Hierzu sind evtl. entsprechende Einstellungen in vorgeschalteten Routern und Firewalls vorzunehmen. Die Worker-Clients müssen lediglich in der Lage sein, ausgehende Verbindungen auf diesem Port aufzubauen, wozu üblicherweise keine gesonderten Einstellungen erforderlich sind. Hierdurch soll eine möglichst unkomplizierte Client-Installation gewährleistet werden.

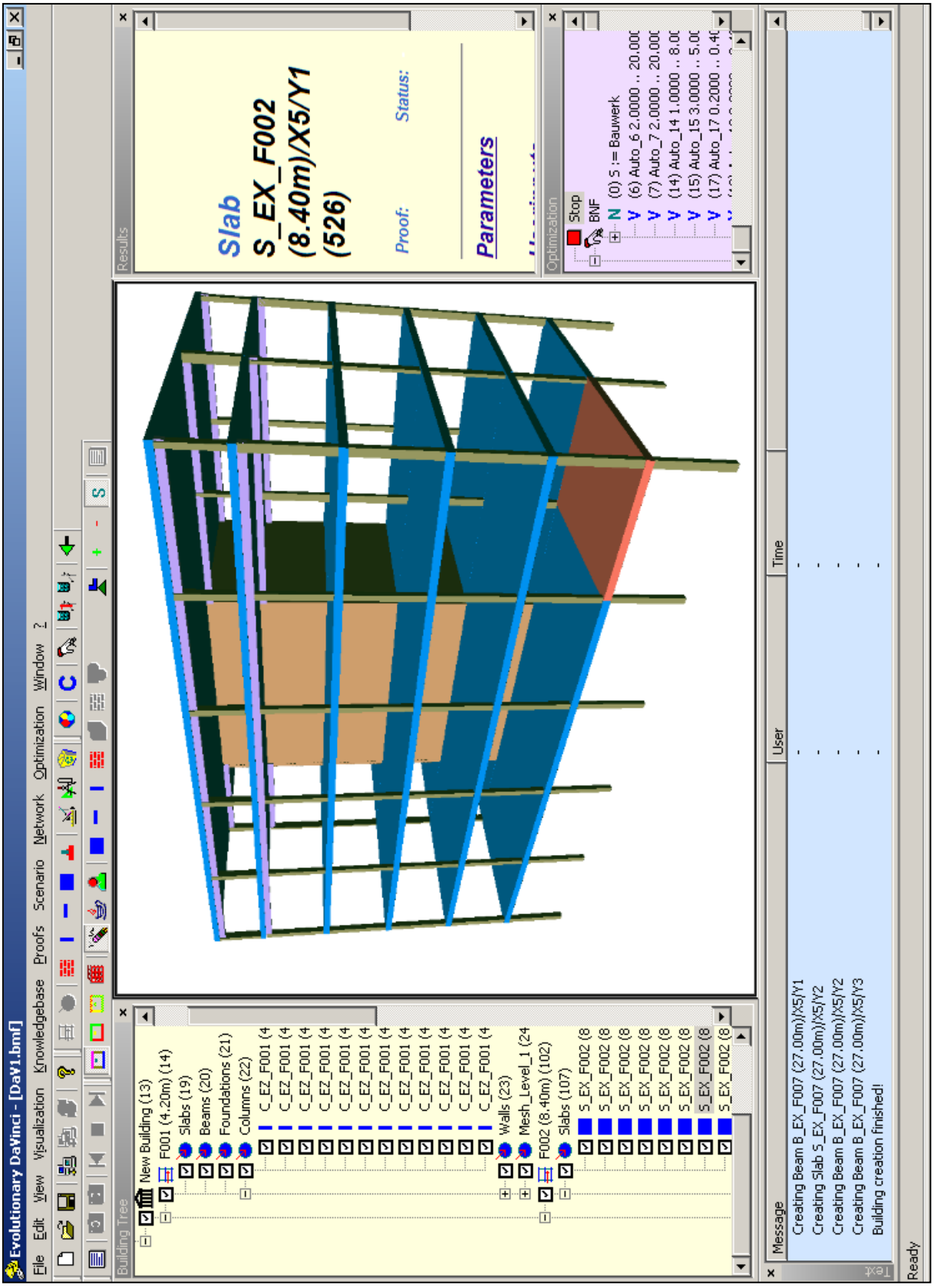


Abbildung 5-10: Implementierung der Applikation „Evolutionary DaVinci“

6 Anwendungsbeispiel

Das in der vorliegenden Arbeit entwickelte System beinhaltet eine Reihe komplexer Komponenten, deren Zusammenspiel sich vorzugsweise an einem praxisnahen Anwendungsbeispiel erläutern lässt. Das im Folgenden dargestellte Beispiel umfasst den vollständigen und auf die zum Verständnis notwendigen Anteile reduzierten Ablauf eines Optimierungsprozesses. Die entwickelten Komponenten werden anhand ihres jeweiligen Einsatzzweckes innerhalb des Optimierungsprozesses umfassend dargestellt. Die geometrischen Randbedingungen des Beispiels werden hinsichtlich des Komplexitätsgrades derart gewählt, dass die zur Anwendung kommenden Verfahrensweisen klar erkennbar sind.

6.1 Vorbereitung

Vor Beginn eines Optimierungsvorgangs ist zunächst detailliert festzulegen, welche Informationen über ein zu optimierendes Bauwerk bereits feststehen, welche hingegen im Rahmen einer Optimierung verändert werden sollen und in welchen Grenzen dies geschehen darf. In der Realität sind zumeist bestimmte Randbedingungen und Vorgaben einzuhalten, die helfen können, den anfänglich sehr großen Suchraum des Optimierungsvorganges einzuschränken. Diese Einschränkungen sind im Rahmen der Optimierung als vorteilhaft anzusehen, da sie die Konzentration der zu optimierenden Parameter auf die möglichen Bereiche fokussieren und dort eine detailliertere Suche nach möglichst optimalen Lösungen unterstützen.

6.1.1 Optimierungs-Vorlage, Randbedingungen

Die Optimierungs-Vorlage dient der Festlegung der zu optimierenden Parameter sowie deren möglicher Struktur. Zusätzlich können hier konstante Parameter festgelegt werden, die in allen Ausprägungen der Individuen identisch vorhanden sein werden.

In dem hier vorgestellten Beispiel soll ein einfaches Hochhaus erstellt werden, das folgende Bedingungen erfüllt:

- Stützenraster mit 3 oder 4 Feldern je Richtung
- Feldgröße zwischen 6 und 12 m
- Bis zu 2 Kerne
- Stockwerke in Gruppen zusammengefasst; beliebige Aufteilung in 1-6 Gruppen mit der Möglichkeit zur Bildung einzelner Outrigger-Stockwerke
- Deckensystem: Flachdecke oder Unterzugsdecke, Ausrichtung x oder y
- Deckenstärke zwischen 20 und 40 cm
- Unterzüge h=20-40cm, b=20-30cm
- Stützen-Abmessungen zwischen 20 und 60 cm
- Anzahl der Stockwerke gesamt: 10

Die gegebenen Randbedingungen werden mit Hilfe der Akquisitionskomponente der Bildungsvorschrift entsprechend Listing 6-1 formalisiert. Die Anzahl der Stockwerke

geht als Konstante in die Wissensbank ein, da diese als feste Größe nicht optimiert wird. Die in Listing 6-1 definierte Variable „Stockwerkgruppe_Ende“ weist eine undefinierte Anzahl von Stockwerken auf. Die Wissensbank interpretiert dies als eine beliebige Anzahl von Stockwerken, die variabel bis zum Erreichen der gewünschten Gesamtzahl von Stockwerken des Gesamtgebäudes angepasst wird.

```
S:=GEB[Raster Kerne Stockwerkgruppen]

Raster:=FELDER_X= Felder ;FELDER_Y= Felder ;RASTER_X= Rastergroesse
;RASTER_Y= Rastergroesse

Felder:=[3;4;1]

Rastergroesse:=[6;12;0.1]

Kern:=KERN[KFELD_X_PRZ= KernfeldX ;KFELD_Y_PRZ= KernfeldY ;KFELDER_X_PRZ=
KernfelderX ;KFELDER_Y_PRZ= KernfelderY ;KMAX_STW= [2;20;1] ;]

KernfeldX:=[0;1;0.01]

KernfeldY:=[0;1;0.01]

KernfelderX:=[0;0.4;0.01]

KernfelderY:=[0;0.4;0.01]

Stockwerkgruppen:=Stockwerkgruppe_Ende|Stockwerkgruppe Stockwerkgrup-
pe_Ende|Stockwerkgruppe Stockwerkgruppe_Outrigger Stockwerkgruppe Stock-
werkgruppe_Ende|Stockwerkgruppe Stockwerkgruppe_Outrigger Stockwerkgruppe
Stockwerkgruppe_Outrigger Stockwerkgruppe Stockwerkgruppe_Ende

Stockwerkgruppe:=SWG[ANZ_STW= [1;6;1] ;H_STW= [3;5;.1] ;OUTRIGGER=2.0;
Deckensystem Ssg1 Ssg2 Ssg3 Ssg4 ]

Stockwerkgruppe_Ende:=SWG[ANZ_STW=0;H_STW= [3;5;.1] ;OUTRIGGER=2.0; De-
ckensystem Ssg1 Ssg2 Ssg3 Ssg4 ]

Stockwerkgruppe_Outrigger:=SWG[ANZ_STW=1;H_STW= [2;4;.2] ;OUTRIGGER=1.0;
Deckensystem Ssg1 Ssg2 Ssg3 Ssg4 ]

Deckensystem:=Fd|Uzd

Fd:=DS=FD;H_D= [0.2;0.4;0.01] ;

Uzd:=DS=UZD;H_D= [0.2;0.4;0.01] ;H_UZ= [0.2;0.4;0.01] ;B_UZ=
[0.2;0.3;0.01] ;DIR_UZ= Dir_Uzd ;

Dir_Uzd:=x|y

Ssg1:=SSG[NAME=x-edge;DX= SsgDim ;DY= SsgDim ;]

Ssg2:=SSG[NAME=y-edge;DX= SsgDim ;DY= SsgDim ;]

Ssg3:=SSG[NAME=corner;DX= SsgDim ;DY= SsgDim ;]

Ssg4:=SSG[NAME=inside;DX= SsgDim ;DY= SsgDim ;]

SsgDim:=[0.20;0.60;0.01]
```

Listing 6-1: Bildungsvorschrift

6.1.2 Parameter der Wissensbank

Weitere in der Wissensbank anzupassende Konstanten sind sämtliche Parameter, die zur späteren Ermittlung der dynamischen und statischen Einwirkungen für die FE-Berechnung und Bemessung erforderlich sind. Hierzu zählen insbesondere orts- und nutzungsabhängige Parameter, die nicht aus vorhandenen Daten ermittelt wer-

den können. Tabelle 6-1 zeigt exemplarisch einige der im vorliegenden Anwendungsbeispiel verwendeten Parameter.

Tabelle 6-1: Parameter der Wissensbank

Kategorie_Nutzung	B	
Erdbebenzone	1	
Untergrundverhaeltnis	A-R	
Bedeutungskategorie	1	
Windzone	1	
q_k	1,5	kN/m²
Duktilitaetsklasse	1	
Daempfung	0,05	
gamma_g	1,35	
gamma_q	1,5	
Kostenindex_Beton	140	€/m³
Kostenindex_Stahl	1.000	€/t

6.1.3 Optimierungsparameter

Die globalen Optimierungsparameter werden in Verbindung mit dem Optimierungs-Template innerhalb der Akquisitionskomponente festgelegt. Die Optimierung verläuft in zwei Phasen, daher werden sämtliche Parameter je Phase getrennt definiert. Die in Abbildung 6-1 gewählten Parameter der ersten Phase erzeugen eine relativ hohe Variantenvielfalt, während im späteren Verlauf geringfügigere Änderungen der Individuen zu einer feineren Optimierung führen sollen. Die Grenze (Generation) zwischen den Phasen wird als „Section Limit“, die Anzahl der Generationen in der Übergangsphase als „Section transition“ angegeben. Vergleiche hierzu auch Kap. 2.1.2.5 - Exploration vs. Exploitation sowie Kap. 4.2.8 - Parameteradaption.

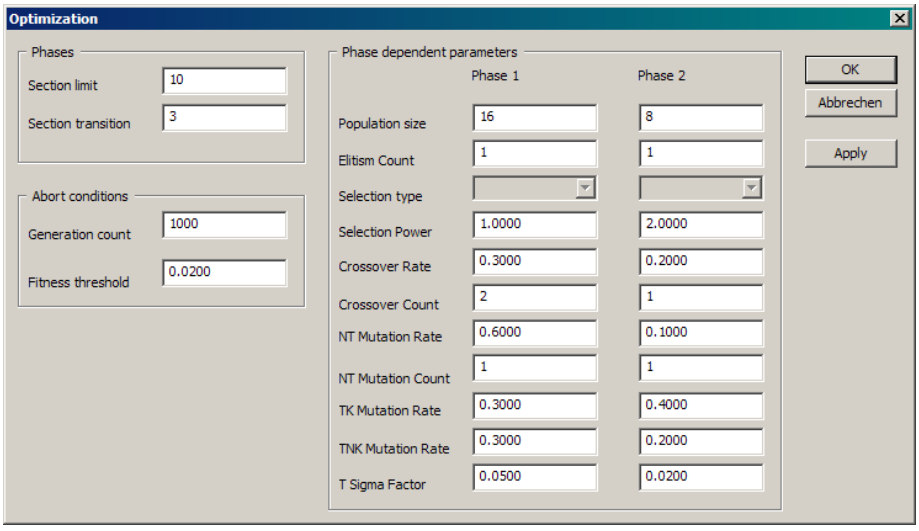


Abbildung 6-1: Optimierungsparameter

6.2 Ablauf des Optimierungsprozesses

Der Optimierungsprozess ist ein iterativer Vorgang, der je nach Umfang der Berechnung einige Stunden oder Tage in Anspruch nehmen kann. In Abbildung 6-2 ist ein grober Überblick der zu durchlaufenden Schritte gegeben.

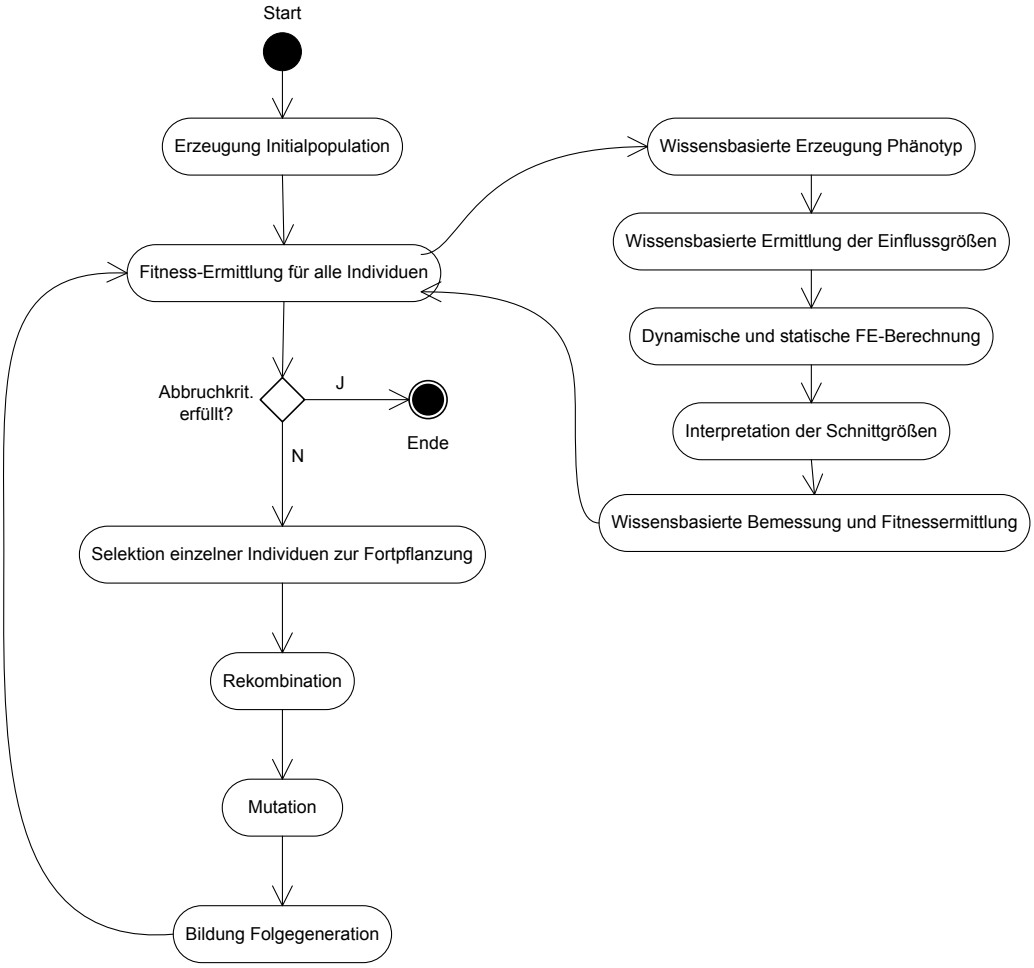
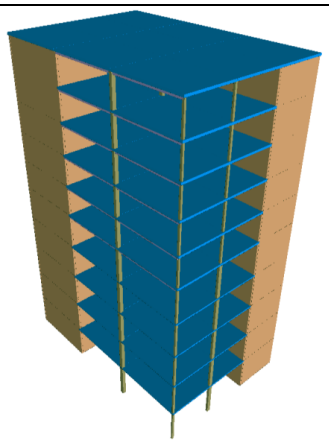
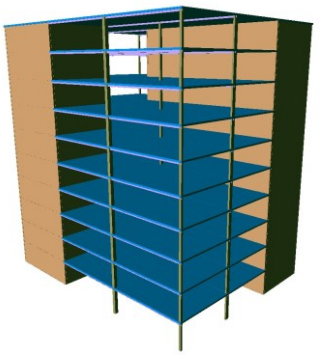


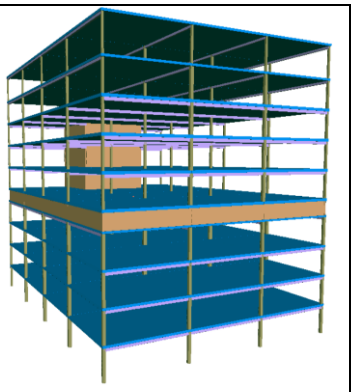
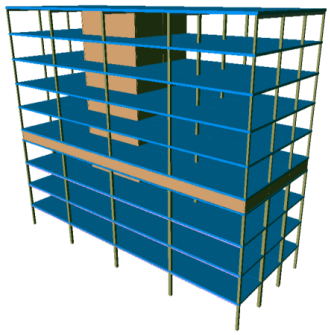
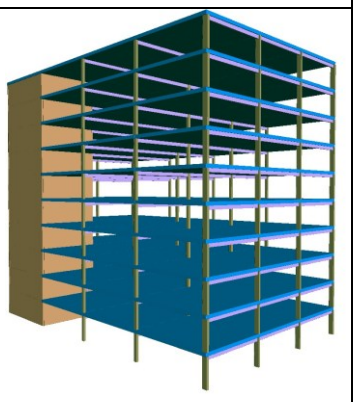
Abbildung 6-2: Ablaufdiagramm des Optimierungsprozesses

6.2.1 Bildung einer Initialpopulation

Die Bildung einer Initialpopulation kann wahlweise zielgerichtet oder zufällig erfolgen. Im vorliegenden Beispiel wurde eine zufällige Erzeugung der Initialpopulation vorgenommen. Hierbei wird anhand der vorgegebenen Populationsgröße eine entsprechende Anzahl von Individuen entsprechend der Bildungsvorschrift erstellt. Die Individuen liegen dabei zunächst nur in ihrer genotypischen Repräsentation vor. Tabelle 6-2 zeigt eine Auflistung der zufällig generierten Individuen der Initialpopulation sowie deren räumliche Visualisierung und Fitnessbewertung. Die Darstellung der Visualisierung sowie der Fitness dienen an dieser Stelle lediglich der Veranschaulichung. Die eigentliche Simulation sowie die Fitnessermittlung findet erst in den folgenden Schritten statt, die in Kapitel 6.2.2. erläutert werden.

Tabelle 6-2: Genotypische Repräsentation von Individuen der Initialpopulation

Nr.	Genotyp	Darstellung und Fitness
1	GEB[FELDER_X=3.00000000;FELDER_Y=3.00000000;RASTER_X=6.30000000;RASTER_Y=9.50000000;KERN[KFELD_X_PRZ=0.04000000;KFELD_Y_PRZ=0.73000000;KFELDER_X_PRZ=0.23000000;KFELDER_Y_PRZ=0.11000000;KMAX_STW=19.00000000;]KERN[KFELD_X_PRZ=0.87000000;KFELD_Y_PRZ=0.02000000;KFELDER_X_PRZ=0.39000000;KFELDER_Y_PRZ=0.25000000;KMAX_STW=17.00000000;]KERN[KFELD_X_PRZ=0.77000000;KFELD_Y_PRZ=0.05000000;KFELDER_X_PRZ=0.40000000;KFELDER_Y_PRZ=0.13000000;KMAX_STW=17.00000000;]SWG[ANZ_STW=5.00000000;H_STW=4.70000000;OUTRIGGER=2.0;DS=FD;H_D=0.21000000;SSG[NAME=x-edge;DX=0.46000000;DY=0.32000000;]SSG[NAME=y-edge;DX=0.45000000;DY=0.51000000;]SSG[NAME=corner;DX=0.28000000;DY=0.24000000;]SSG[NAME=inside;DX=0.41000000;DY=0.44000000;]]SWG[ANZ_STW=0;H_STW=4.00000000;OUTRIGGER=2.0;DS=UZD;H_D=0.28000000;H_UZ=0.26000000;B_UZ=0.21000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.26000000;DY=0.24000000;]SSG[NAME=y-edge;DX=0.46000000;DY=0.39000000;]SSG[NAME=corner;DX=0.22000000;DY=0.28000000;]SSG[NAME=inside;DX=0.32000000;DY=0.52000000;]]]	 F=0.256368
2	GEB[FELDER_X=3.00000000;FELDER_Y=3.00000000;RASTER_X=11.20000000;RASTER_Y=8.50000000;KERN[KFELD_X_PRZ=0.66000000;KFELD_Y_PRZ=0.86000000;KFELDER_X_PRZ=0.21000000;KFELDER_Y_PRZ=0.03000000;KMAX_STW=7.00000000;]KERN[KFELD_X_PRZ=0.14000000;KFELD_Y_PRZ=0.11000000;KFELDER_X_PRZ=0.00000000;KFELDER_Y_PRZ=0.28000000;KMAX_STW=7.00000000;]KERN[KFELD_X_PRZ=0.19000000;KFELD_Y_PRZ=0.13000000;KFELDER_X_PRZ=0.31000000;KFELDER_Y_PRZ=0.12000000;KMAX_STW=10.00000000;]KERN[KFELD_X_PRZ=0.78000000;KFELD_Y_PRZ=0.85000000;KFELDER_X_PRZ=0.33000000;KFELDER_Y_PRZ=0.24000000;KMAX_STW=17.00000000;]SWG[ANZ_STW=0;H_STW=3.60000000;OUTRIG-GER=2.0;DS=UZD;H_D=0.21000000;H_UZ=0.20000000;B_UZ=0.22000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.43000000;DY=0.45000000;]SSG[NAME=y-edge;DX=0.22000000;DY=0.53000000;]SSG[NAME=corner;DX=0.30000000;DY=0.27000000;]SSG[NAME=inside;DX=0.25000000;DY=0.32000000;]]]	 F=1.076473
3	GEB[FELDER_X=4.00000000;FELDER_Y=3.00000000;RASTER_X=12.00000000;RASTER_Y=7.70000000;KERN[KFELD_X_PRZ=0.27000000;KFELD_Y_PRZ=0.13000000;KFELDER_X_PRZ=0.04000000;KFELDER_Y_PRZ=0.09000000;KMAX_STW=20.00000000;]KERN[KFELD_X_PRZ=0.61000000;KFELD_Y_PRZ=0.82000000;KFELDER_X_PRZ=0.08000000;KFELDER_Y_PRZ=0.09000000;KMAX_STW=12.00000000;]KERN[KFELD_X_PRZ=0.97000000;KFELD_Y_PRZ=0.85000000;KFELDER_X_PRZ=0.30000000;KFELDER_Y_PRZ=0.08000000;KMAX_STW=7.00000000;]SWG[ANZ_STW=3.00000000;H_STW=3.10000000;OUTRIGGER=2.0;DS=UZD;H_D=0.32000000;H_UZ=0.29000000;B_UZ=0.26000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.37000000;DY=0.59000000;]SSG[NAME=y-edge;DX=0.57000000;DY=0.58000000;]SSG[NAME=corner;DX=0.53000000;DY=0.21000000;]SSG[NAME=inside;DX=0.27000000;DY=0.35000000;]]SWG[ANZ_STW=0;H_STW=3.80000000;OUTRIGGER=2.0;DS=UZD;H_D=0.38000000;H_UZ=0.29000000;B_UZ=0.25000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.43000000;DY=0.26000000;]SSG[NAME=y-edge;DX=0.44000000;DY=0.20000000;]SSG[NAME=corner;DX=0.53000000;DY=0.36000000;]SSG[NAME=inside;DX=0.24000000;DY=0.50000000;]]]	 F=2.943293
4	GEB[FELDER_X=3.00000000;FELDER_Y=4.00000000;RASTER_X=10.60000000;RASTER_Y=10.70000000;KERN[KFELD_X_PRZ=0.07000000;KFELD_Y_PRZ=0.27000000;KFELDER_X_PRZ=0.00000000;KFELDER_Y_PRZ=0.06000000;KMAX_STW=20.00000000;]KERN[KFELD_X_PRZ=0.42000000;KFELD_Y_PRZ=0.83000000;KFELDER_X_PRZ=0.26000000;KFELDER_Y_PRZ=0.18000000;KMAX_STW=13.00000000;]SWG[ANZ_STW=4.00000000;H_STW=4.00000000;OUTRIGGER=2.0;DS=UZD;H_D=0.29000000;H_UZ=0.34000000;B_UZ=0.29000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.37000000;DY=0.35000000;]SSG[NAME=y-edge;DX=0.42000000;DY=0.46000000;]SSG[NAME=corner;DX=0.35000000;DY=0.34000000;]SSG[NAME=inside;DX=0.5000000;DY=0.4100000;]]SWG[ANZ_STW=1;H_STW=2.20000000;OUTRIGGER=1.0;DS=UZD;H_D=0.23000000;H_UZ=0.35000000;B_UZ=0.21000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.23000000;DY=0.59000000;]SSG[NAME=y-edge;DX=0.57000000;DY=0.48000000;]SSG[NAME=corner;DX=0.48000000;DY=0.55000000;]SSG[NAME=inside;DX=0.42000000;DY=0.45000000;]]SWG[ANZ_STW=4.00000000;H_STW=3.90000000;OUTRIGGER=2.0;DS=U	

	ZD;H_D=0.34000000;H_UZ=0.37000000;B_UZ=0.24000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.37000000;DY=0.40000000;]SSG[NAME=y-edge;DX=0.30000000;DY=0.31000000;]SSG[NAME=corner;DX=0.26000000;DY=0.32000000;]SSG[NAME=inside;DX=0.34000000;DY=0.27000000;]]SWG[ANZ_STW=0;H_STW=4.40000000;OUTRIGGER=2.0;DS=UZD;H_D=0.27000000;H_UZ=0.28000000;B_UZ=0.27000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.25000000;DY=0.28000000;]SSG[NAME=y-edge;DX=0.21000000;DY=0.33000000;]SSG[NAME=corner;DX=0.24000000;DY=0.27000000;]SSG[NAME=inside;DX=0.34000000;DY=0.31000000;]]]	 F=20.464951
5	GEB[FELDER_X=4.00000000;FELDER_Y=3.00000000;RASTER_X=11.60000000;RASTER_Y=6.10000000;KERN[KFELD_X_PRZ=0.29000000;KFELD_Y_PRZ=0.76000000;KFELDER_X_PRZ=0.28000000;KFELDER_Y_PRZ=0.12000000;KMAX_STW=13.00000000;]KERN[KFELD_X_PRZ=0.37000000;KFELD_Y_PRZ=0.34000000;KFELDER_X_PRZ=0.35000000;KFELDER_Y_PRZ=0.30000000;KMAX_STW=19.00000000;]SWG[ANZ_STW=4.00000000;H_STW=4.40000000;OUTRIGGER=2.0;DS=UZD;H_D=0.28000000;H_UZ=0.24000000;B_UZ=0.28000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.48000000;DY=0.40000000;]SSG[NAME=y-edge;DX=0.43000000;DY=0.55000000;]SSG[NAME=corner;DX=0.35000000;DY=0.38000000;]SSG[NAME=inside;DX=0.48000000;DY=0.44000000;]]SWG[ANZ_STW=1;H_STW=2.00000000;OUTRIGGER=1.0;DS=FD;H_D=0.25000000;SSG[NAME=x-edge;DX=0.31000000;DY=0.39000000;]SSG[NAME=y-edge;DX=0.57000000;DY=0.54000000;]SSG[NAME=corner;DX=0.41000000;DY=0.29000000;]SSG[NAME=inside;DX=0.42000000;DY=0.41000000;]]SWG[ANZ_STW=5.00000000;H_STW=4.20000000;OUTRIGGER=2.0;DS=FD;H_D=0.30000000;SSG[NAME=x-edge;DX=0.38000000;DY=0.34000000;]SSG[NAME=y-edge;DX=0.32000000;DY=0.27000000;]SSG[NAME=corner;DX=0.26000000;DY=0.31000000;]SSG[NAME=inside;DX=0.33000000;DY=0.46000000;]]SWG[ANZ_STW=0;H_STW=3.90000000;OUTRIGGER=2.0;DS=UZD;H_D=0.35000000;H_UZ=0.38000000;B_UZ=0.22000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.28000000;DY=0.29000000;]SSG[NAME=y-edge;DX=0.25000000;DY=0.28000000;]SSG[NAME=corner;DX=0.21000000;DY=0.23000000;]SSG[NAME=inside;DX=0.34000000;DY=0.37000000;]]]	 F=3.107712
6	GEB[FELDER_X=3.00000000;FELDER_Y=4.00000000;RASTER_X=8.20000000;RASTER_Y=10.00000000;KERN[KFELD_X_PRZ=0.05000000;KFELD_Y_PRZ=0.83000000;KFELDER_X_PRZ=0.21000000;KFELDER_Y_PRZ=0.29000000;KMAX_STW=20.00000000;]SWG[ANZ_STW=0;H_STW=3.50000000;OUTRIGGER=2.0;DS=UZD;H_D=0.38000000;H_UZ=0.39000000;B_UZ=0.21000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.22000000;DY=0.56000000;]SSG[NAME=y-edge;DX=0.30000000;DY=0.40000000;]SSG[NAME=corner;DX=0.29000000;DY=0.59000000;]SSG[NAME=inside;DX=0.34000000;DY=0.39000000;]]]	 F=6.540665

6.2.2 Ermittlung der Fitness

Die in einer Generation der Optimierung enthaltenen Individuen werden einzeln anhand der Fitness-Funktion bewertet. Die Fitness-Funktion beinhaltet sämtliche zur Bewertung eines Bauwerks erforderlichen Schritte, deren Bestimmung den weitaus größten Berechnungsaufwand des Optimierungsprozesses ausmacht. In den folgenden Unterkapiteln wird schrittweise die Ermittlung der Fitness eines beliebigen Individuums dargestellt. Als Beispiel wird in diesem Fall Individuum Nr. 1 der Initialpopula-

tion (Tabelle 6-2) verwendet. Die in den folgenden Unterkapiteln beschriebenen Schritte beziehen sich lediglich auf die Bewertung dieses Individuums und werden jeweils erneut zur Bewertung der weiteren Individuen durchlaufen.

6.2.2.1 Interpretation der genotypischen Bauwerksbeschreibung

Die genotypische Beschreibung des Individuums wird zunächst in ihrer Struktur analysiert und das enthaltene Faktenwissen extrahiert. In Tabelle 6-3 sind hierzu einige Beispiele dargestellt.

Tabelle 6-3: Interpretation des Genotypen

Beschreibung	**Interpretation**
`GEB[`	Beginn der Gebäudebeschreibung

`FELDER_X=3.00000000;`
`FELDER_Y=3.00000000;`
`RASTER_X=6.30000000;`
`RASTER_Y=9.50000000;`
	Das Stützenraster soll in x- und y-Richtung jeweils drei Felder aufweisen. Die Spannweiten betragen 6,3m für x bzw. 9,5m für y

`KERN[`
`KFELD_X_PRZ=0.04000000;`
`KFELD_Y_PRZ=0.73000000;`
`KFELDER_X_PRZ=0.23000000;`
`KFELDER_Y_PRZ=0.11000000;`
`KMAX_STW=19.00000000;]`
	Definition eines Kerns. Die Lage innerhalb des Rasters soll bei x=4% (->erstes Feld) und y=73% (->drittes Feld) sein. Die Abmessungen sind x=23% und y=11% des Rasters, was durch die geringe Anzahl an Feldern jeweils ein Feld ergibt. Der Kern soll maximal 19 Stockwerke hoch sein (Hierbei darf jedoch nicht die Gesamtzahl der Stockwerke des Bauwerks, in diesem Beispiel 10, überschritten werden).

`KERN...`	Definition eines weiteren Kerns
`SWG[ANZ_STW=5.00000000;H_STW=4.70000000;OUTRIGGER=false;DS=FD;H_D=0.21000000;`	Definition einer Stockwerkgruppe. Diese umspannt fünf Stockwerke mit einer Höhe von jeweils 4,7m. Diese Stockwerke enthalten keine Outrigger-Aussteifung. Als Deckensystem kommen Flachdecken der Stärke 21cm zum Einsatz.
`SSG[NAME=corner;DX=0.28000000;DY=0.24000000;]`	Definition einer Stützensystemgruppe innerhalb der Stockwerkgruppe. Anhand des Namens „corner“ werden die Eckstützen definiert, die einen Querschnitt von 28*24cm aufweisen sollen.
`SSG[NAME=inside;DX=0.41000000;DY=0.44000000;]]]`	Eine weitere Gruppe definiert die Innenstützen. Nicht dargestellt ist die Definition der Randstützen.
`SWG[ANZ_STW=0;H_STW=4.00000000;OUTRIG-` `GER=2.0;DS=UZD;H_D=0.28000000;H_UZ=0.26000000;B_UZ=0.21000000;DIR_UZ=y;SSG[…];SSG[…]]`	Definition einer weiteren Stockwerkgruppe ohne Festlegung der Stockwerksanzahl. Diese Gruppe wird zum „Auffüllen“ der insgesamt benötigten Stockwerksanzahl herangezogen.
`]`	Ende der Gebäudebeschreibung

6.2.2.2 Erzeugung des Topologiemodells

Das zuvor gewonnene Faktenwissen wird zur Erzeugung eines topologischen Modells verwendet, das in Abbildung 6-3 dargestellt ist. Die Grundrissgestaltung entspricht der im Genotypen definierten Rastergeometrie. Die vertikale Anordnung der Stockwerkbereiche orientiert sich an Gesamthöhe und Anzahl der Stockwerke. Mehrere Stockwerke können hierbei zu Stockwerkgruppen zusammengefasst sein.

Die Anzahl der Stockwerke einer Stockwerkgruppe kann neben der Definition innerhalb des Genotypen auch durch Berechnung am Gesamtsystem ermittelt werden, beispielsweise wenn eine maximale Bauwerkshöhe oder eine fixe Gesamtanzahl an Geschossen vorgegeben ist. In diesem Fall ist die Definition der obersten Stockwerkgruppe ohne Angabe der Geschosszahl sinnvoll.

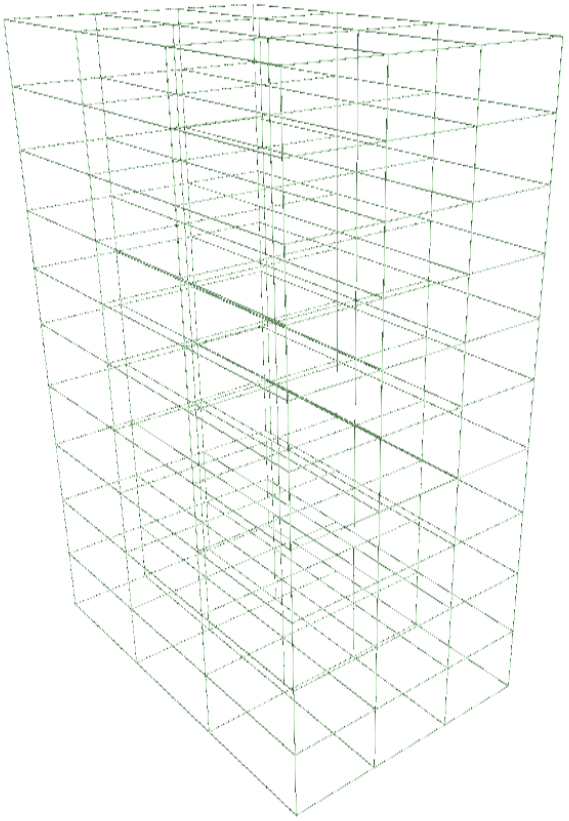


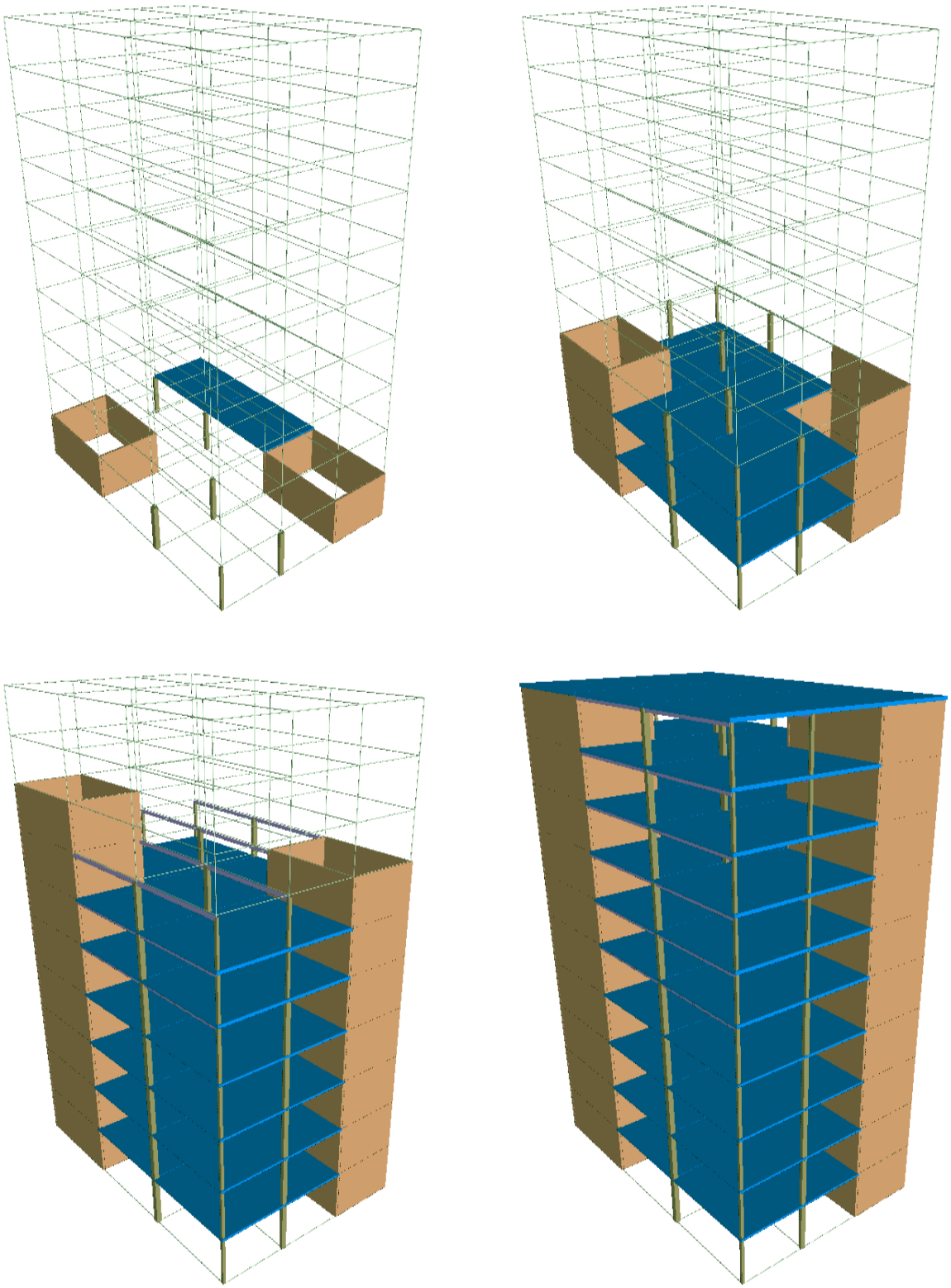
Abbildung 6-3: Topologiemodell

6.2.2.3 Erzeugung des Strukturmodells

Basierend auf den Achsen des topologischen Modells erfolgt die Erzeugung einzelner Bauteile zur Bildung des Strukturmodells. Anhand von wissensbasiert formalisierten Regeln wird hierbei für jede Kante des Topologiemodells ermittelt, welche Bauteile an der betreffenden Stelle zu erzeugen sind und wie deren weitere Parameter (z.B. Abmessungen) definiert sind.

Die Regeln und Berechnungselemente der Wissensbank können hierbei auf Konstanten, Tabellenwerte, Ergebnisse interner Berechnungen oder Daten der genotypischen Beschreibung zurückgreifen. Einige Phasen dieses Vorganges sind in Abbildung 6-4 dargestellt, wobei die letzte Phase das endgültige Strukturmodell darstellt. Eine Übersicht der verwendeten Wissensbank ist dem Anhang zu entnehmen.

Abbildung 6-4: Schrittweise Erzeugung des Strukturmodells



6.2.2.4 Erzeugung des FE-Netzes

Nach der Erstellung des Strukturmodells erfolgt die Transformation des Modells in ein ganzheitliches Finite-Elemente-Modell. Der Grad der Granularität des FE-Netzes je Richtung wird anhand wissensbasiert formalisierter Formeln ermittelt. Als Wand- oder Deckenteil wird hier jeweils ein Strukturelement angesehen, dass sich zwischen den Gebäudeachsen erstreckt. In dem hier dargestellten Beispiel werden die Strukturelemente der Stützen und Balken durch je sechs Balkenelemente, Wand- und Deckenteile durch je 6*6=36 Schalenelemente repräsentiert. Eine Deckenplatte des nebenstehenden Bauteils besteht somit aus 3*3=9 Strukturelementen, die jeweils durch 36 FE-Elemente abgebildet werden. Insgesamt besteht alleine die obere Deckenplatte aus 324 Schalenelementen.

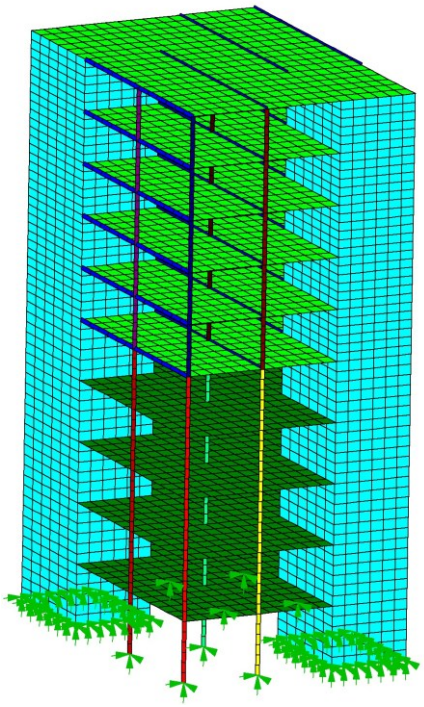


Abbildung 6-5: FE-Modell in B&B

Das beschriebene Finite-Elemente Modell wird nach Ermittlung der Einwirkungen zusammen mit diesen an das externe System *Antras B&B* übergeben. Abbildung 6-5 zeigt die Darstellung des FE-Netzes in B&B. Die farbliche Hervorhebung stellt unterschiedliche Querschnittsgruppen dar. Im vorliegenden Beispiel sind zwei unterschiedliche Stockwerkgruppen ausgebildet, wobei neben veränderten Stützenquerschnitten auch die Deckensysteme unterschiedlich ausgebildet wurden (oben: Unterzugdecken, unten: Flachdecken).

6.2.2.5 Dynamische FE-Berechnung

Basierend auf dem FE-Modell wird zunächst eine B&B Eingabedatei zur dynamischen Berechnung generiert. Die Eingabedatei enthält neben den Knoten, Elementen, Kopplungen, Material- und Querschnittswerten auch Zusatzmassen zur Repräsentation der abgeminderten Verkehrslast auf Decken. Zusätzlich werden die notwendigen Steuergrößen für die Eigenwertanalyse sowie das Antwortspektrenverfahren generiert.

Ein Auszug der B&B Eingabedatei zur dynamischen Berechnung ist in Listing 6-2 dargestellt. Die Originaldatei enthält 6784 Knoten und 6240 Elemente. Aufgrund der Systemarchitektur des Programms B&B ist es erforderlich, jeweils für X- und Y-Richtung eine eigene Eingabedatei zu erzeugen.

Die Ermittlung der Steuergrößen erfolgt mittels in der Wissensbank hinterlegter Formeln und Tabellen, die anhand global definierter Parameter (siehe Kap. 6.1.2) ausgewertet werden. Als Beispiel dient die in der Wissensbank formalisierte Tabelle zu Parametern des Untergrundverhältnisses (Abbildung 6-6) [DIN4149]. Die Angabe des Parameters „Untergrundverhaeltnis=A-R“ ermöglicht die Ermittlung der Parameter S_h, T_B_h, T_C_h, T_D_h zur Beschreibung des Antwortspektrums.

```
   Projektdaten:
   ============================
 10 Gebaeudeoptimierung Phaenotyp

   Steuergrößen:
   ============================
 13 9 0 1 s kN m 0 0 0 0 0 0 0 0

   Knoten:
   ============================
 23 0 0 1 0.000000 0.000000 0.000000
 23 0 0 2 0.000000 0.000000 0.783333
...
 23 0 0 6784 -81.099998 26.916668 43.500000

   Freiheitsgrade:
   ============================
 24 1 0 0 0 0 0 0
...
 24 6784 0 0 0 0 0 0

   Materialien:
   ============================
 30 0 1 28300000.000000 0.200000 0.000000 0 0 0 0 0
 30 0 2 28300000.000000 0.200000 25.000000 0 0 0 0 0

   Querschnitte:
   ============================
 31 0 1 0.064600 0.000428 0.000777 0.000156 0.043067 0.043067
...
 32 8 0.0000 0.0000 0.270000 0.0000 0.0000 0.270000

   Gruppenzuordnung:
   ============================
 36 1 1 1
...
 36 6240 2 8

   Elemente:
   ============================
 37 0 111 1 1 2 3
 37 0 111 2 2 4 5
 ...
 37 0 111 6240 6736 6142 6784

-------- Bis hier identisch für statische Berechnung ------------

   Zusatzmasse für dyn. Berechnung
   ============================
   Abgeminderte Verkehrslast auf Decken:
 44 1 3 337 0.000 0.000 6.129861
...
 44 1 3 6192 0.000 0.000 6.129861

   Steuergrößen Dynamik Erdbebenuntersuchung
   ============================
 61 0 0 2 9.81

   Steuergrößen Eigenwertanalyse
   ============================
 62 10 10 100 0.010000 1

   Steuergrößen Erdbebeneinwirkung in x-Richtung
   ============================
 65 2 1 0 1 0 1 0 0 0.400000 1.000000 0.800000 2.500000 1.500000 0.050000 0.200000 2.000000 1
    1 0.050000 0
```

Listing 6-2: B&B Eingabedatei zur dynamischen Berechnung (x-Richtung)

Komplex Table

Table axles

Axle X (results) | Axle Y (input values) | Check | OK

Untergrundverhaeltnis / ->	S_h	T_B_h	T_C_h	T_D_h
A-R	1.0000	0.0500	0.2000	2.0000
B-R	1.2500	0.0500	0.2500	2.0000
C-R	1.5000	0.0500	0.3000	2.0000
B-T	1.0000	0.1000	0.3000	2.0000
C-T	1.2500	0.1000	0.4000	2.0000
C-S	0.7500	0.1000	0.5000	2.0000

Abbildung 6-6: Tabelle „Untergrundverhältnis“

Nach erfolgter Erzeugung der Eingabedateien erfolgt der Aufruf des externen B&B Berechnungskerns. Die Ergebnisse der dynamischen Berechnung enthalten die Eigenformen und Eigenfrequenzen des Tragwerks sowie die modalen Knotenlasten aus dem Antwortspektrenverfahren. Die B&B Visualisierungskomponente „BuBView“ erlaubt eine komfortable Visualisierung der Berechnungsergebnisse. In Abbildung 6-7 sind exemplarisch einige Eigenformen dargestellt.

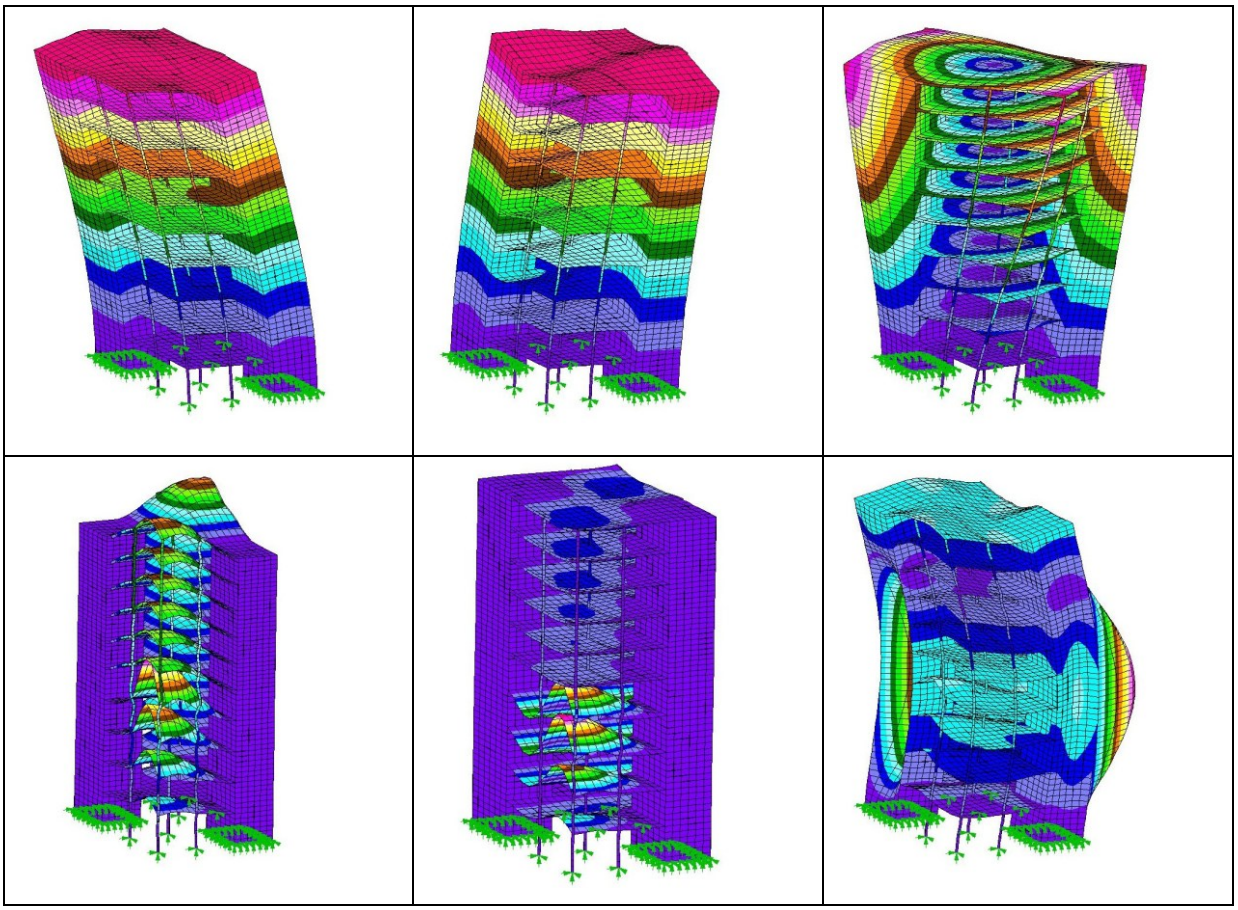


Abbildung 6-7: Ausgewählte Eigenformen, Ergebnisse der dynamischen FE-Berechnung

6.2.2.6 Statische FE-Berechnung

Nach Abschluss der dynamischen Berechnung erfolgt der Import der Eigenfrequenzen sowie der modalen Ersatzlasten aus der Erdbebenuntersuchung. Innerhalb des Entwurfssystems werden die statisch erforderlichen Lastfälle und Lastfallkombinationen erzeugt. Die hierfür notwendigen Parameter (Materialkennwerte, Verkehrslasten abhängig von der Nutzung, Kombinationsbeiwerte etc.) werden wie bei der dynamischen Berechnung aus wissensbasiert definierten Formeln und Tabellen unter Anwendung der global definierten Optimierungsparameter gewonnen.

Die Ermittlung der Windlasten erfolgt gemäß DIN 1055-4 [DIN1055-4]. Auch die hierfür notwendigen Formeln sind vollständig wissensbasiert formalisiert. Nach DIN 1055-4 ist es notwendig, die Windlasten je Richtung ausmittig anzusetzen, wodurch sich zunächst vier Lastfälle ergeben, die jeweils positiv und negativ anzusetzen sind. In Verbindung mit Eigengewicht und Verkehrslast ist die Auswertung von 24 Lastfallkombinationen erforderlich.

```
-------- Beginn siehe Listing 6-2 ------------

Lastfall: Ständige Lasten
 43 1 1 0.0 0.0 -1.0 1 1 1 1
 43 1 2 0.0 0.0 -1.0 2 1 2 1
...
 43 1 6240 0.0 0.0 -1.0 6240 1 6240 1

Lastfall: Verkehrslasten
 40 2 1 12 0.000000 0.000000 -1.532465
...

Lastfall: Windlast X1
 40 3 1 12 68.686371 0.000000 0.000000
...

Lastfall: Windlast Y1
 40 4 1 12 0.000000 47.170631 0.000000
...

Lastfall: Windlast X2
 40 5 1 12 68.686371 0.000000 0.000000
...

Lastfall: Windlast Y2
 40 6 1 12 0.000000 47.170631 0.000000
...

Lastfall: Modallasten X Mode 01
 40 7 1 12 -0.003860 -0.000078 -0.000414
...

Lastfall: Modallasten X Mode 02
 40 8 1 12 -0.000004 0.000128 0.000015
...

Lastfall: Modallasten X Mode 03
 40 9 1 12 -0.000096 -0.000029 0.002789
...

Lastfall: Modallasten Y Mode 01
 40 10 1 12 -0.000117 -0.000002 -0.000013
...

Lastfall: Modallasten Y Mode 02
 40 11 1 12 0.000134 -0.003949 -0.000452
...

Lastfall: Modallasten Y Mode 03
 40 12 1 12 -0.000042 -0.000013 0.001216
...

Lastfallkombinationen:
```

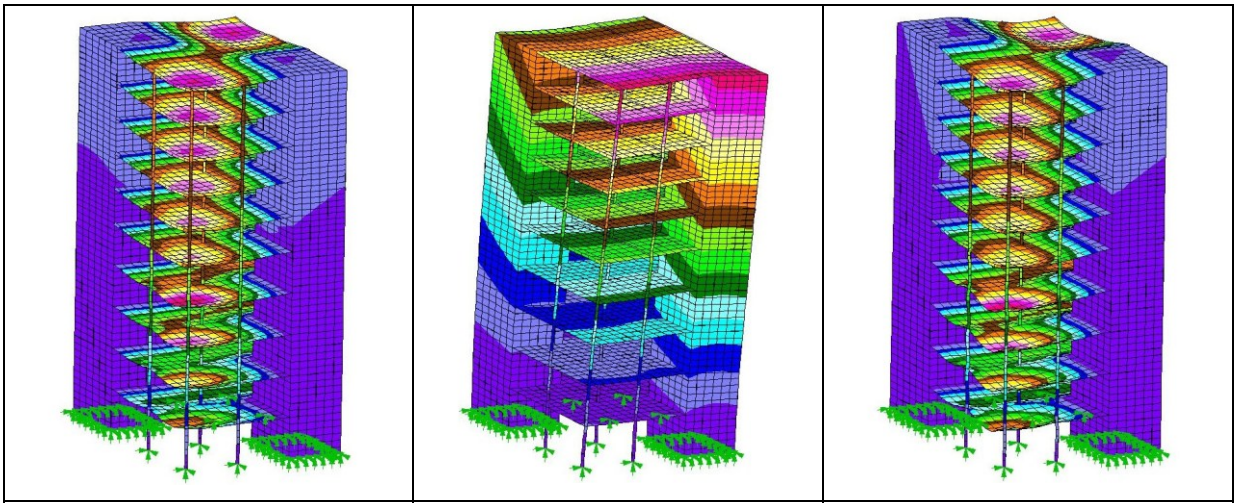
```
Ständig
49 13 1 1.350000

Ständig + Verkehr
49 14 1 1.350000 2 1.500000

Ständig + Wind
49 15 1 1.350000 3 1.500000
49 16 1 1.350000 3 -1.500000
49 17 1 1.350000 5 1.500000
49 18 1 1.350000 5 -1.500000
49 19 1 1.350000 4 1.500000
49 20 1 1.350000 4 -1.500000
49 21 1 1.350000 6 1.500000
49 22 1 1.350000 6 -1.500000

Ständig + Verkehr + Wind
49 23 1 1.350000 2 1.500000 3 0.900000
49 24 1 1.350000 2 1.500000 3 -0.900000
49 25 1 1.350000 2 1.500000 5 0.900000
49 26 1 1.350000 2 1.500000 5 -0.900000
49 27 1 1.350000 2 1.000000 3 1.500000
49 28 1 1.350000 2 1.000000 3 -1.500000
49 29 1 1.350000 2 1.000000 5 1.500000
49 30 1 1.350000 2 1.000000 5 -1.500000
49 31 1 1.350000 2 1.500000 4 0.900000
49 32 1 1.350000 2 1.500000 4 -0.900000
49 33 1 1.350000 2 1.500000 6 0.900000
49 34 1 1.350000 2 1.500000 6 -0.900000
49 35 1 1.350000 2 1.000000 4 1.500000
49 36 1 1.350000 2 1.000000 4 -1.500000
49 37 1 1.350000 2 1.000000 6 1.500000
49 38 1 1.350000 2 1.000000 6 -1.500000
```

Listing 6-3: B&B Eingabedatei zur statischen FE-Berechnung



LF1: Ständig	LF3: Wind in X-Richtung	LF31: Ständig + Verkehr + Wind in Y-Richtung

Abbildung 6-8: Verformungsbilder aus Ergebnissen der statischen FE-Berechnung

Die Modallasten der dynamischen Berechnung können unverändert als Lastfälle in der statischen Berechnung übernommen werden. Die Kombination der Lastfälle erfolgt nach der CQC-Regel (Kap. 5.4), wodurch eine einfache Linearkombination der Lastfälle nicht möglich ist. Daher erfolgt die Erzeugung entsprechender Lastfallkombinationen nicht in B&B, sondern nach Abschluss der externen statischen Berechnung innerhalb des Entwurfssystems.

In Listing 6-3 ist ein Auszug aus der B&B Eingabedatei zur statischen Berechnung dargestellt. Es ist zu beachten, dass der erste Teil der Eingabedatei die Strukturdaten des Tragwerks enthält und daher mit der Eingabedatei aus der dynamischen Berechnung (Listing 6-2) identisch ist. Dieser Teil wird in Listing 6-3 nicht mehr dargestellt. Die Kombination der Lastfälle aus Modallasten gem. CQC-Regel erfolgt systembedingt nicht in B&B, sondern erst in einer späteren Phase innerhalb der Optimierungskomponente.

6.2.2.7 Bemessung

Die maximalen Schnittgrößen als Ergebnis der statischen FE-Berechnung werden zur wissensbasierten Bemessung jedes einzelnen Bauteiles auf Ebene des Strukturmodells herangezogen. Exemplarisch sind in Tabelle 6-4 die Ergebnisse der wissensbasierten Bemessung eines Deckenelementes im achten Obergeschoss dargestellt. Die Ergebnisse beinhalten neben den reinen Bemessungsergebnissen auch die zur Bewertung notwendigen Materialmengen sowie einen Straffaktor zur Berücksichtigung nicht erfüllter Nachweise. Ein ausführlicheres Bemessungsbeispiel ist in Kap. 5.5 zu finden.

Tabelle 6-4: Ergebnisse der wissensbasierten Bemessung eines Deckenelementes

Parameter	*Wert*	*Einh.*
Q_DECKE	3,50	kN/m²
q_k	3,50	kN/m²
Nutzungsart	Büro	
E_Modul	28.300.000	kN/m²
Betonfestigkeitsklasse	C30/37	
Nue	0,20	-
Gamma	25,00	-
Gamma_Dyn	25,00	-
Flaeche	63,05	m²
dx	6,50	m
dy	9,70	m
Straffaktor	0,0000	-
mue_Eds_rechnerisch	0,0671	-
mue_Eds_lim	0,296	-
M_Eds	74,224	kNm
b	1,00	m
d	25,50	cm
f_cd	17,00	N/mm²
Stress_M11_Min	74,224	kNm
Stress_M22_Min	64,872	kNm
Stress_M11_Max	45,164	kNm
Stress_M22_Max	45,045	kNm
h	0,28	m
c_nom	20,0	mm
d_sl_1	10,0	mm
h_Decke	0,28	m
alpha_C	0,85	-
f_ck	30,00	N/mm²
gamma_c_Ortbeton	1,50	-
c_min	10	mm
delta_c	10	mm

Expositionsklasse	XC1	
Grundflaeche	63,05	m²
Konstruktionsflaeche	0,00	m²
Volumen	17,65	m³
dz	0,28	m
erf_A_sl	6,6231	cm²/m
Volumen_Stahl	0,0835	m³
sigma_s1_d	456,50	N/mm²
omega_1	0,0697	-
N_Ed	0,00	kN
mue_Eds	0,0671	-

6.2.2.8 *Ermittlung des Fitnesswertes*

Die Ermittlung des Fitnesswertes (Bewertungsindex) bezogen auf das Gesamtbauwerk erfolgt durch Akkumulation der Materialmassen und Straffaktoren aller bemessenen Strukturbauteile. Die Ermittlung des Bewertungsindex wurde in Kapitel 5.6.2 erläutert. Tabelle 6-5 enthält eine Zusammenstellung der ermittelten Werte zur Bestimmung des Fitnesswertes.

Tabelle 6-5: Zusammenstellung der Fitnesswertermittlung

Parameter	*Wert*	*Einh.*
Summe der Materialkosten	332.577,81	€
Baukostenindex	4,0	-
Gesamtkosten	*1.330.311,25*	*€*
Summe der Deckenflächen	4.442,80	m²
Summe der Konstruktionsflächen	161,82	m²
Nutzfläche	4.280,98	m²
Ertragsindex	1.600,00	€/m²
Gesamtertrag	*6.849.568,00*	*€*
Summe der Straffaktoren	*0,32*	
Bewertung (Fitness)	**0,2564**	

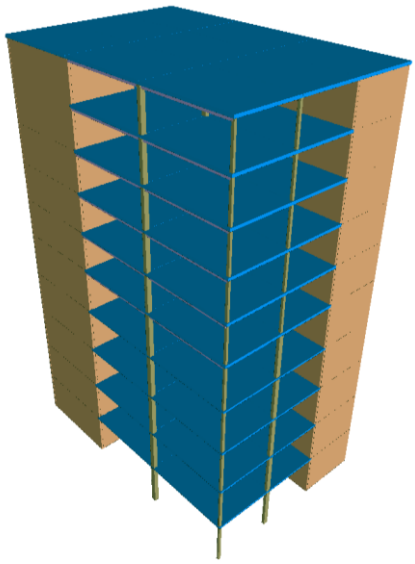
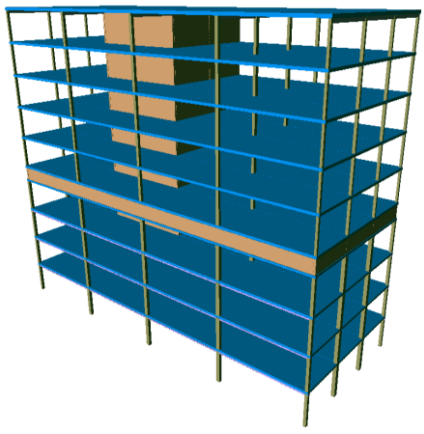
6.2.3 Bildung einer Folgegeneration

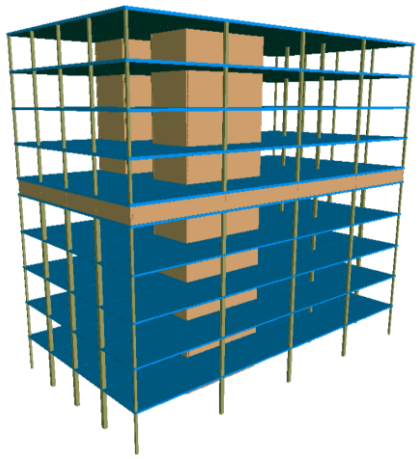
Nach Abschluss der Bewertung sämtlicher Individuen einer Generation erfolgt die Selektion von Individuen zur Bildung einer weiteren Generation. Eine Elitismus-Funktion stellt sicher, dass zunächst die *n* besten Individuen (in vorliegendem Beispiel: *n*=1) in eine Folgegeneration übernommen werden, um zu vermeiden, dass während des Optimierungsvorganges eine Verschlechterung des jeweils bisher gefundenen besten Individuums eintritt. Zusätzlich kommen die genetischen Operationen Rekombination und Mutation entsprechend der in den Optimierungsparametern definierten Wahrscheinlichkeiten zum Einsatz.

Ein anschauliches Beispiel für eine Rekombination ist in Tabelle 6-6 dargestellt. In diesem Beispiel wurden die genetischen Informationen der Individuen 1 und 5 der Initialpopulation miteinander gekreuzt. Als Basis des neuen Individuums dient Individuum 5. Die Kreuzung erfolgt am Startknoten der unteren Stockwerkgruppe von Ind. 5. Diese wird gegen die untere Stockwerkgruppe aus Individuum 1 ausgetauscht. Die

Visualisierung offenbart deutlich die Veränderung in Bezug auf die Anzahl der Stockwerke, Stützengeometrie und Art des gewählten Deckensystems. Zusätzlich zum Vorgang der Kreuzung erfolgt eine Mutation der Anzahl der Raster-Felder in Y-Richtung. Die genetische Information der Kerne wurde nicht verändert. Trotzdem erfolgt aufgrund der neuen Rastergeometrie eine Teilung der beiden Gebäudekerne, die zuvor in einem großen Kern vereinigt waren.

Tabelle 6-6: Kreuzungs-Operation

Nr.	Genotyp	Darstellung und Fitness
Elter 1 (Ind. 1)	GEB[FELDER_X=3.00000000;FELDER_Y=3.00000000;RASTER_X=6.30000000;RASTER_Y=9.50000000;KERN[KFELD_X_PRZ=0.04000000;KFELD_Y_PRZ=0.73000000;KFELDER_X_PRZ=0.23000000;KFELDER_Y_PRZ=0.11000000;KMAX_STW=19.00000000;]KERN[KFELD_X_PRZ=0.87000000;KFELD_Y_PRZ=0.02000000;KFELDER_X_PRZ=0.39000000;KFELDER_Y_PRZ=0.25000000;KMAX_STW=17.00000000;]KERN[KFELD_X_PRZ=0.77000000;KFELD_Y_PRZ=0.05000000;KFELDER_X_PRZ=0.40000000;KFELDER_Y_PRZ=0.13000000;KMAX_STW=17.00000000;]**SWG[ANZ_STW=5.00000000;H_STW=4.70000000;OUTRIGGER=2.0;DS=FD;H_D=0.21000000;SSG[NAME=x-edge;DX=0.46000000;DY=0.32000000;]SSG[NAME=y-edge;DX=0.45000000;DY=0.51000000;]SSG[NAME=corner;DX=0.28000000;DY=0.24000000;]SSG[NAME=inside;DX=0.41000000;DY=0.44000000;]]**SWG[ANZ_STW=0;H_STW=4.00000000;OUTRIGGER=2.0;DS=UZD;H_D=0.28000000;H_UZ=0.26000000;B_UZ=0.21000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.26000000;DY=0.51000000;]SSG[NAME=y-edge;DX=0.46000000;DY=0.39000000;]SSG[NAME=corner;DX=0.37000000;DY=0.42000000;]SSG[NAME=inside;DX=0.32000000;DY=0.52000000;]]]	 F=0.256368
Elter 2 (Ind. 5)	GEB[FELDER_X=4.00000000;FELDER_Y=3.00000000;RASTER_X=11.60000000;RASTER_Y=6.10000000;KERN[KFELD_X_PRZ=0.29000000;KFELD_Y_PRZ=0.76000000;KFELDER_X_PRZ=0.28000000;KFELDER_Y_PRZ=0.12000000;KMAX_STW=13.00000000;]KERN[KFELD_X_PRZ=0.37000000;KFELD_Y_PRZ=0.34000000;KFELDER_X_PRZ=0.35000000;KFELDER_Y_PRZ=0.30000000;KMAX_STW=19.00000000;]*SWG[ANZ_STW=4.00000000;H_STW=4.40000000;OUTRIGGER=2.0;DS=UZD;H_D=0.28000000;H_UZ=0.24000000;B_UZ=0.28000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.48000000;DY=0.40000000;]SSG[NAME=y-edge;DX=0.43000000;DY=0.55000000;]SSG[NAME=corner;DX=0.35000000;DY=0.38000000;]SSG[NAME=inside;DX=0.48000000;DY=0.44000000;]]*SWG[ANZ_STW=1;H_STW=2.00000000;OUTRIGGER=1.0;DS=FD;H_D=0.25000000;SSG[NAME=x-edge;DX=0.31000000;DY=0.39000000;]SSG[NAME=y-edge;DX=0.57000000;DY=0.54000000;]SSG[NAME=corner;DX=0.41000000;DY=0.29000000;]SSG[NAME=inside;DX=0.42000000;DY=0.41000000;]]SWG[ANZ_STW=5.00000000;H_STW=4.20000000;OUTRIGGER=2.0;DS=FD;H_D=0.30000000;SSG[NAME=x-edge;DX=0.38000000;DY=0.34000000;]SSG[NAME=y-edge;DX=0.32000000;DY=0.27000000;]SSG[NAME=corner;DX=0.26000000;DY=0.31000000;]SSG[NAME=inside;DX=0.33000000;DY=0.46000000;]]SWG[ANZ_STW=0;H_STW=3.90000000;OUTRIGGER=2.0;DS=UZD;H_D=0.35000000;H_UZ=0.38000000;B_UZ=0.22000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.28000000;DY=0.29000000;]SSG[NAME=y-edge;DX=0.25000000;DY=0.28000000;]SSG[NAME=corner;DX=0.21000000;DY=0.23000000;]SSG[NAME=inside;DX=0.34000000;DY=0.37000000;]]]	 F=3.107712

Nach-komme	GEB[FELDER_X=4.00000000;FELDER_Y=4.00000000;RASTER_X=11.60000000;RASTER_Y=6.10000000;KERN[KFELD_X_PRZ=0.29000000;KFELD_Y_PRZ=0.76000000;KFELDER_X_PRZ=0.28000000;KFELDER_Y_PRZ=0.12000000;KMAX_STW=13.00000000;]KERN[KFELD_X_PRZ=0.37000000;KFELD_Y_PRZ=0.34000000;KFELDER_X_PRZ=0.35000000;KFELDER_Y_PRZ=0.30000000;KMAX_STW=19.00000000;]**SWG[ANZ_STW=5.00000000;H_STW=4.70000000;OUTRIGGER=2.0;DS=FD;H_D=0.21000000;SSG[NAME=x-edge;DX=0.46000000;DY=0.32000000;]SSG[NAME=y-edge;DX=0.45000000;DY=0.51000000;]SSG[NAME=corner;DX=0.28000000;DY=0.24000000;]SSG[NAME=inside;DX=0.41000000;DY=0.44000000;]]**SWG[ANZ_STW=1;H_STW=2.00000000;OUTRIGGER=1.0;DS=FD;H_D=0.25000000;SSG[NAME=x-edge;DX=0.31000000;DY=0.39000000;]SSG[NAME=y-edge;DX=0.57000000;DY=0.54000000;]SSG[NAME=corner;DX=0.41000000;DY=0.29000000;]SSG[NAME=inside;DX=0.42000000;DY=0.41000000;]]SWG[ANZ_STW=5.00000000;H_STW=4.20000000;OUTRIGGER=2.0;DS=FD;H_D=0.30000000;SSG[NAME=x-edge;DX=0.38000000;DY=0.34000000;]SSG[NAME=y-edge;DX=0.32000000;DY=0.27000000;]SSG[NAME=corner;DX=0.26000000;DY=0.31000000;]SSG[NAME=inside;DX=0.33000000;DY=0.46000000;]]SWG[ANZ_STW=0;H_STW=3.90000000;OUTRIGGER=2.0;DS=UZD;H_D=0.35000000;H_UZ=0.38000000;B_UZ=0.22000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.28000000;DY=0.29000000;]SSG[NAME=y-edge;DX=0.25000000;DY=0.28000000;]SSG[NAME=corner;DX=0.21000000;DY=0.23000000;]SSG[NAME=inside;DX=0.34000000;DY=0.37000000;]]]	 F= 3.033672

Anhand des Fitnesswertes des Nachkommen ist erkennbar, dass Ind. 5 durch die Veränderung der unteren Stockwerkgruppe geringfügig verbessert wurde. Trotzdem fällt unter den gegebenen Randbedingungen die Bewertung von Ind. 1 wesentlich günstiger aus. Sowohl Ind. 5 als auch der daraus generierte Nachkomme weisen aufgrund der zwei Kerne eine unnötig hohe Konstruktionsfläche auf. Gleichzeitig wird durch das statisch nicht erforderliche Outrigger-Geschoss die Nutzfläche weiter reduziert. Aufgrund der geringen Deckenstärken bei großen Spannweiten in x-Richtung sind die Nachweise der Decken nicht vollständig erfüllt, was sich in der Gesamtbewertung durch einen enormen Anstieg der Straffaktoren bemerkbar macht.

Nachdem durch Anwendung der genetischen Operationen Selektion, Rekombination und Mutation eine vollständige neue Generation erzeugt wurde, wird der Optimierungszyklus mit der Bewertung der neuen Individuen fortgesetzt.

6.2.4 Abschluss der Optimierung

Nach Erreichen eines der Abbruchkriterien (Fitness unterhalb eines gewünschten Schwellwertes bzw. erreichte Anzahl an Generationen) gilt die Optimierung als abgeschlossen. Ebenso kann der Optimierungsvorgang jederzeit manuell angehalten oder abgebrochen werden. Die Abbruchbedingungen sind Teil der zu Beginn gewählten Optimierungsparameter (siehe Abbildung 6-1).

Im vorliegenden Beispiel erfolgt der Abbruch nach 1000 Generationen, oder wenn der Fitnesswert von 0,02 unterschritten wurde. Die sich zu diesem Zeitpunkt in der letzten vollständig bewerteten Generation befindlichen Individuen mit dem besten Fitnesswert stellen das Ergebnis der Optimierung dar. Da in dem vorliegenden Beispiel eine einkriterielle Optimierung erfolgte, kann eindeutig das Individuum mit der besten Fitness bestimmt werden. Die Ergebnisse des beschriebenen Optimierungsprozesses sowie der Optimierungsverlauf sind in Kap. 7.2 dargestellt.

7 Experimentelle Versuchsreihen

Die im Folgenden dargestellten Ergebnisse aus experimentellen Versuchsreihen dienen der Evaluation und Verifikation der entwickelten Optimierungsplattform. Im Rahmen der vorliegenden Arbeit kann lediglich ein geringer Teil der potentiellen Möglichkeiten der Optimierungsplattform dargestellt werden. Die dargestellten Experimente sollen die grundsätzliche Arbeitsweise und Wirksamkeit der Optimierungsplattform demonstrieren.

Es wurden drei Experimente unterschiedlicher Komplexitätsgrade durchgeführt. Hierbei kamen jeweils die gleichen Wissensbanken für Entwurf und Bewertung zum Einsatz. Die einzige Abweichung bestand jeweils in unterschiedlichen Definitionsbereichen der Grundriss-Raster sowie in der unterschiedlichen Anzahl an Stockwerken.

*7.1 Experiment 1: Tragwerk mit 4 Stockwerken, Raster 1*1 Feld*

Das erste Experiment umfasst die Optimierung eines sehr kleinen Bauwerks. Das Grundraster besteht aus einem einzelnen Feld, die vertikale Ausprägung des Bauwerks erstreckt sich über vier Stockwerke. Dieses sehr einfache Experiment wurde gewählt, da es aufgrund der geringen Komplexität eine sehr schnelle Fitness-Berechnung ermöglicht und somit zur Überprüfung und Anpassung von Optimierungsparametern und zur Verifikation des gesamten Optimierungsprozesses dienen kann. Aufgrund der geringen Anzahl an Strukturelementen ergibt sich jedoch kein repräsentatives Anwendungsszenario zur Demonstration des Optimierungsprozesses. Hierzu dienen die nachfolgenden Experimente.

```
S:=Bauwerk

Bauwerk:=GEB[ Raster Kerne Stockwerkgruppen ]

Raster:=FELDER_X= Felder ;FELDER_Y= Felder ;RASTER_X= Rastergroesse
;RASTER_Y= Rastergroesse ;

Felder:=[1;1;1]

Rastergroesse:=[5;9;0.05]

Kerne:=|Kern

KernfeldX:=[0;1;0.01]

KernfeldY:=[0;1;0.01]

KernfelderX:=[0;0.4;0.01]

KernfelderY:=[0;0.4;0.01]

Kern:=KERN[KFELD_X_PRZ= KernfeldX ;KFELD_Y_PRZ= KernfeldY ;KFELDER_X_PRZ=
KernfelderX ;KFELDER_Y_PRZ= KernfelderY ;KMAX_STW= [2;20;1] ;]

Stockwerkgruppen:=Stockwerkgruppe_Ende|Stockwerkgruppe Stockwerkgrup-
pe_Ende|Stockwerkgruppe Stockwerkgruppe_Outrigger Stockwerkgruppe Stock-
werkgruppe_Ende|Stockwerkgruppe Stockwerkgruppe_Outrigger Stockwerkgruppe
Stockwerkgruppe_Outrigger Stockwerkgruppe Stockwerkgruppe_Ende

Stockwerkgruppe:=SWG[ANZ_STW= [1;6;1] ;H_STW= [3;5;.1] ;OUTRIGGER=2.0;
Deckensystem Ssg1 Ssg2 Ssg3 Ssg4 ]

Stockwerkgruppe_Ende:=SWG[ANZ_STW=0;H_STW= [3;5;.1] ;OUTRIGGER=2.0; De-
ckensystem Ssg1 Ssg2 Ssg3 Ssg4 ]

Stockwerkgruppe_Outrigger:=SWG[ANZ_STW=1;H_STW= [2;4;.2] ;OUTRIGGER=1.0;
Deckensystem Ssg1 Ssg2 Ssg3 Ssg4 ]
```

```
Deckensystem:=Fd|Uzd

Fd:=DS=FD;H_D=[0.2;0.4;0.01];

Uzd;DS=UZD;H_D= [0.2;0.4;0.01] ;H_UZ= [0.2;0.4;0.01] ;B_UZ=
[0.2;0.3;0.01] ;DIR_UZ= Dir_Uzd ;

Dir_Uzd:=x|y

Ssg1:=SSG[NAME=x-edge;DX= SsgDim ;DY= SsgDim ;]

Ssg2:=SSG[NAME=y-edge;DX= SsgDim ;DY= SsgDim ;]

Ssg3:=SSG[NAME=corner;DX= SsgDim ;DY= SsgDim ;]

Ssg4:=SSG[NAME=inside;DX= SsgDim ;DY= SsgDim ;]

SsgDim:=[0.20;0.60;0.01]
```

Listing 7-1: Bildungsvorschrift

```
Phase 1:
Phase duration = 10                 Anzahl der Generationen für Phase 1
Population Size = 16                Populationsgröße
Selection Type = 0                  Selektionsart: Rouletterad-
                                    Selektion
Selection Power = 1                 Selektionsdruck 1
Elitism Count = 1                   Anzahl der Elitismus-Individuen
Crossover Count = 2                 Anzahl Kreuzungsknoten (falls Kreu-
                                    zung erfolgt)
Crossover Rate = .3                 Kreuzungsrate
Mutation NT Count = 1               Anzahl Mutationsknoten nicht-
                                    terminierter Symbole (falls Mutati-
                                    on erfolgt)
Mutation NT Rate = .6               Mutationsrate nicht-terminierter
                                    Symbole
Mutation TK Rate = .3               Mutationsrate terminierter, kardi-
                                    naler Symbole
Mutation TNK Rate = .3              Mutationsrate terminierter, nicht
                                    kardinaler Symbole
Factor Sigma = 0.05                 Globaler Faktor für die Streuung
                                    (Sigma) bei Mutation numerischer
                                    Symbole sowie kardinaler terminier-
                                    ter Symbole

Phase 1-2 transition: 3             Anzahl der Übergangsgenerationen
                                    zwischen Phasen 1 und 2

Phase 2:
Population Size= 8
Selection Type = 0
Selection Power = 2
Elitism Count = 1
```

```
Crossover Count = 1
Crossover Rate = 0.2
Mutation NT Count = 1
Mutation NT Rate = 0.1
Mutation TK Rate = 0.4
Mutation TNK Rate = 0.2
Factor Sigma = 0.02
```

Listing 7-2: Optimierungsparameter Experiment 1

7.1.1 Optimierungsversuch 1

Der erste Optimierungsversuch verwendet in der ersten Optimierungsphase eine Populationsgröße von 16 Individuen, die in der zweiten Phase nach 10 Generationen auf 8 Individuen reduziert wird. Die weiteren Optimierungsparameter sind in Listing 7-2 dargestellt. Der Versuch wurde zur Verifikation mehrmals unabhängig durchgeführt. Abbildung 7-1 zeigt den Optimierungsverlauf der jeweils besten Individuen sowie den gemittelten Optimierungsverlauf aus drei Versuchsläufen. Deutlich zu erkennen ist hier der große Einfluss der zufällig erzeugten Initialpopulation auf den Optimierungsverlauf.

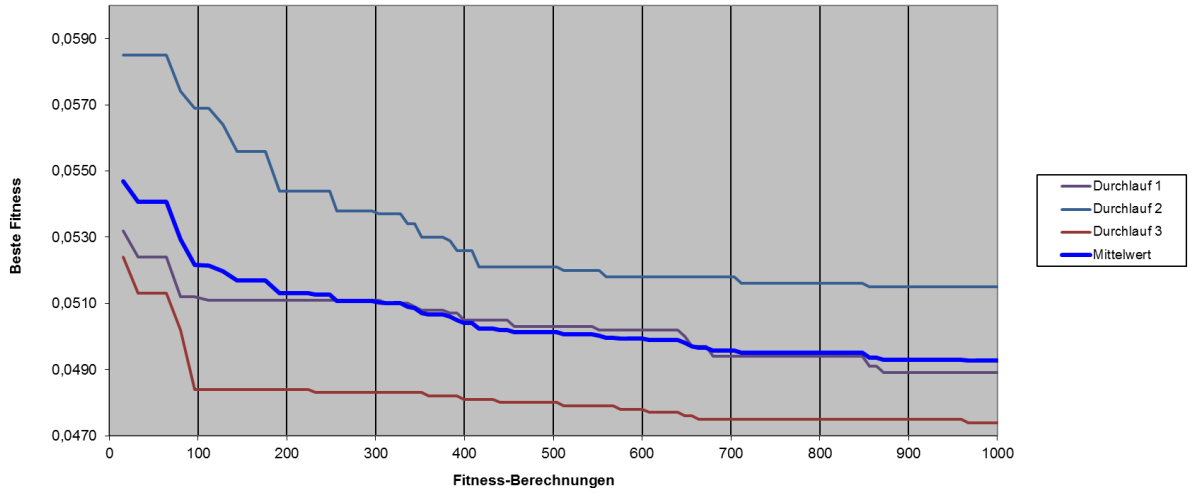


Abbildung 7-1: Optimierungsverlauf Versuch 1, bestes Individuum

7.1.2 Optimierungsversuch 2

Ein weiterer Versuch wurde mit den gleichen Randbedingungen, jedoch unter Veränderung der Populationsgrößen durchgeführt. Die Populationsgrößen bei diesem Versuch sind 30 (Phase 1) bzw. 15 (Phase 2). Die gesteigerte Populationsgröße zu Beginn erlaubt eine größere Varianz und führt dadurch schnell zu besseren konzeptuellen Designs, als dies in Optimierungsversuch 1 der Fall war.

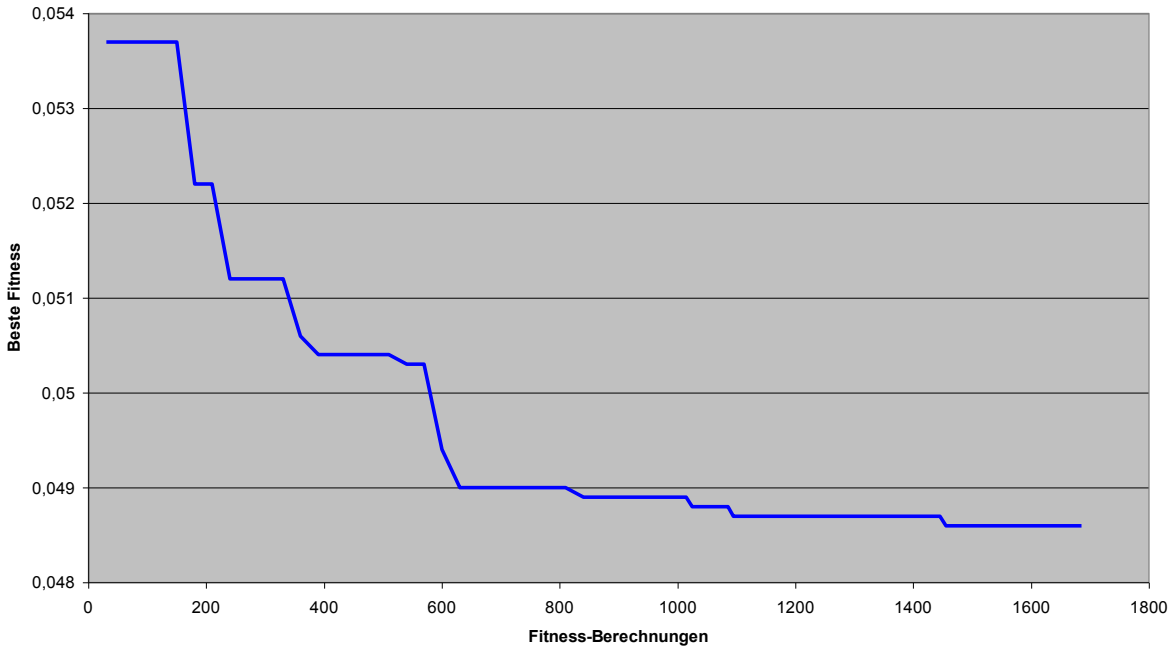


Abbildung 7-2: Optimierungsverlauf Versuch 2, bestes Individuum

Das beste Individuum dieses Versuchslaufes ist exemplarisch in Listing 7-3 sowie Abbildung 7-3 dargestellt. Erwartungsgemäß hat sich bei diesem Individuum eine Unterzugsdecke mit unterschiedlichen Spannweiten je Achse herausgebildet, wobei sich die größere Spannweite in der durch Unterzüge überspannten Richtung befindet. Insgesamt konnte dieses Bauwerk keine besonders großen Feldgrößen erreichen, da bedingt durch das kleine Raster keine aussteifenden Kerne ausgebildet werden können und die gesamte Aussteifungswirkung durch Einspannung der Stützen erreicht wird.

```
GEB[
      FELDER_X=1.00000000;    FELDER_Y=1.00000000;
      RASTER_X=6.00000000;    RASTER_Y=7.20000000;
      SWG[
            ANZ_STW=4.00000000; H_STW=3.20000000; OUTRIGGER=N;
            DS=UZD; H_D=0.20000000;
            H_UZ=0.20000000; B_UZ=0.20000000; DIR_UZ=y;
            SSG[  NAME=x-edge;      DX=0.21000000;    DY=0.50000000;]
            SSG[  NAME=y-edge;      DX=0.47000000;    DY=0.58000000;]
            SSG[  NAME=corner;      DX=0.20000000;    DY=0.32000000;]
            SSG[  NAME=inside;      DX=0.27000000;    DY=0.33000000;]
      ]
  ]
```

Listing 7-3: Beispiel eines Individuums

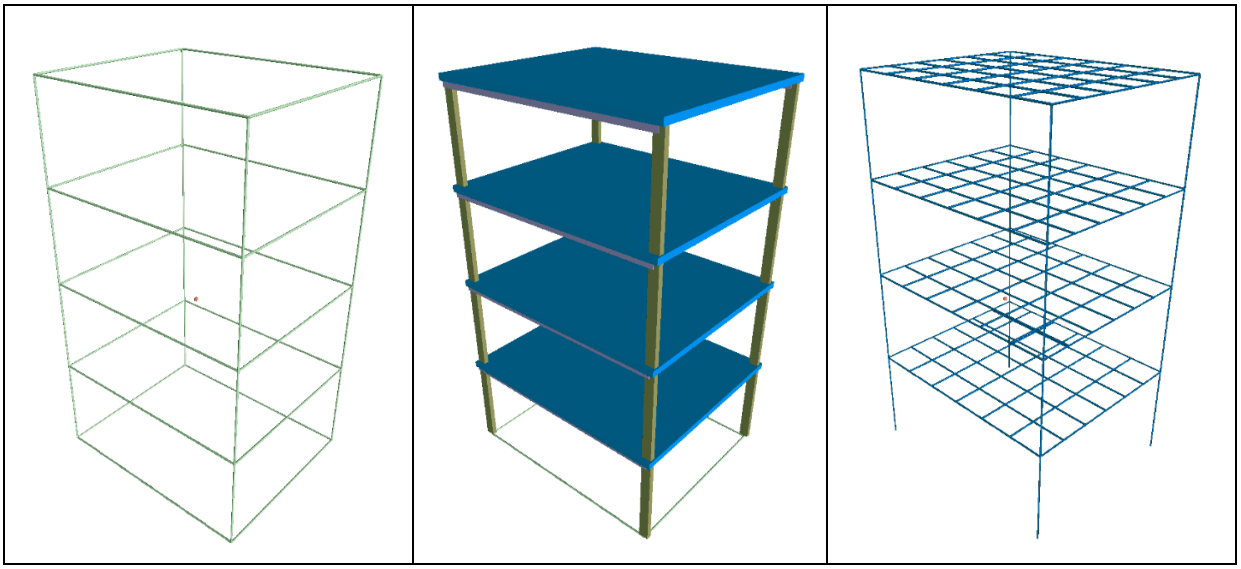


Abbildung 7-3:Topologie- , Struktur- und FE-Modell des Beispiel-Individuums

7.2 Experiment 2: Tragwerk mit 10 Stockwerken, Raster (3-4)(3-4) Felder*

Das in diesem Experiment zu optimierende Bauwerk weist gegenüber dem Bauwerk aus Experiment 1 ein geringfügig vergrößertes Grundraster auf, das je Achse eine Ausprägung zwischen drei und vier Feldern annehmen kann. Dies ermöglicht bereits die Ausbildung von einem oder mehreren Gebäudekernen. Das in Kapitel 6 beschriebene Anwendungsbeispiel basiert vollständig auf diesem Experiment.

```
S:=Bauwerk

Bauwerk:=GEB[ Raster Kerne Stockwerkgruppen ]

Raster:=FELDER_X= Felder ;FELDER_Y= Felder ;RASTER_X= Rastergroesse
;RASTER_Y= Rastergroesse ;

Felder:=[3;4;1]

Rastergroesse:=[5;9;0.05]

Kerne:=|Kern

KernfeldX:=[0;1;0.01]

KernfeldY:=[0;1;0.01]

KernfelderX:=[0;0.4;0.01]

KernfelderY:=[0;0.4;0.01]

Kern:=KERN[KFELD_X_PRZ= KernfeldX ;KFELD_Y_PRZ= KernfeldY ;KFELDER_X_PRZ=
KernfelderX ;KFELDER_Y_PRZ= KernfelderY ;KMAX_STW= [2;20;1] ;]

Stockwerkgruppen:=Stockwerkgruppe_Ende|Stockwerkgruppe Stockwerkgrup-
pe_Ende|Stockwerkgruppe Stockwerkgruppe_Outrigger Stockwerkgruppe Stock-
werkgruppe_Ende|Stockwerkgruppe Stockwerkgruppe_Outrigger Stockwerkgruppe
Stockwerkgruppe_Outrigger Stockwerkgruppe Stockwerkgruppe_Ende

Stockwerkgruppe:=SWG[ANZ_STW= [1;6;1] ;H_STW= [3;5;.1] ;OUTRIGGER=2.0;
```

```
Deckensystem Ssg1 Ssg2 Ssg3 Ssg4 ]

Stockwerkgruppe_Ende:=SWG[ANZ_STW=0;H_STW= [3;5;.1] ;OUTRIGGER=2.0; De-
ckensystem Ssg1 Ssg2 Ssg3 Ssg4 ]

Stockwerkgruppe_Outrigger:=SWG[ANZ_STW=1;H_STW= [2;4;.2] ;OUTRIGGER=1.0;
Deckensystem Ssg1 Ssg2 Ssg3 Ssg4 ]

Deckensystem:=Fd|Uzd

Fd:=DS=FD;H_D=[0.2;0.4;0.01];

Uzd;DS=UZD;H_D= [0.2;0.4;0.01] ;H_UZ= [0.2;0.4;0.01] ;B_UZ=
[0.2;0.3;0.01] ;DIR_UZ= Dir_Uzd ;

Dir_Uzd:=x|y

Ssg1:=SSG[NAME=x-edge;DX= SsgDim ;DY= SsgDim ;]

Ssg2:=SSG[NAME=y-edge;DX= SsgDim ;DY= SsgDim ;]

Ssg3:=SSG[NAME=corner;DX= SsgDim ;DY= SsgDim ;]

Ssg4:=SSG[NAME=inside;DX= SsgDim ;DY= SsgDim ;]

SsgDim:=[0.20;0.60;0.01]
```

Listing 7-4: Bildungsvorschrift Experiment 2

Phase 1:	
Phase duration = 10	Anzahl der Generationen für Phase 1
Population Size = 30	Populationsgröße
Selection Type = 0	Selektionsart: Rouletterad-Selektion
Selection Power = 1	Selektionsdruck 1
Elitism Count = 1	Anzahl der Elitismus-Individuen
Crossover Count = 2	Anzahl Kreuzungsknoten (falls Kreuzung erfolgt)
Crossover Rate = .3	Kreuzungsrate
Mutation NT Count = 1	Anzahl Mutationsknoten nicht-terminierter Symbole (falls Mutation erfolgt)
Mutation NT Rate = .6	Mutationsrate nicht-terminierter Symbole
Mutation TK Rate = .3	Mutationsrate terminierter, kardinaler Symbole
Mutation TNK Rate = .3	Mutationsrate terminierter, nicht kardinaler Symbole
Factor Sigma = 0.05	Globaler Faktor für die Streuung (Sigma) bei Mutation numerischer Symbole sowie kardinaler terminierter Symbole
Phase 1-2 transition: 3	Anzahl der Übergangsgenerationen zwischen Phasen 1 und 2
Phase 2:	

```
Population Size= 15
Selection Type = 0
Selection Power = 2
Elitism Count = 1
Crossover Count = 1
Crossover Rate = 0.2
Mutation NT Count = 1
Mutation NT Rate = 0.1
Mutation TK Rate = 0.4
Mutation TNK Rate = 0.2
Factor Sigma = 0.02
```

Listing 7-5: Optimierungsparameter Experiment 2

Bedingt durch die überschaubaren Dimensionen des hier vorgestellten Beispiels stellt sich als beste Lösung ein symmetrischer Grundriss mit einem einzelnen Kern ein (Tabelle 7-1). Entsprechend ihrer geringeren Belastung sind die Eckstützen schlanker ausgebildet als die Randstützen. Ein weiteres Bauwerk der 100. Generation erreicht eine bessere Aussteifung durch zwei Kerne. Die Anordnung der Kerne erzeugt eine geringere Steifigkeit in X-Richtung, die in diesem Bauwerk durch biegesteif an die Kerne angeschlossene Balken ausgeglichen wird. Die geringe verbleibende Nutzfläche und die höheren Kosten ergeben jedoch eine wesentlich schlechtere Bewertung als diejenige des besten Individuums.

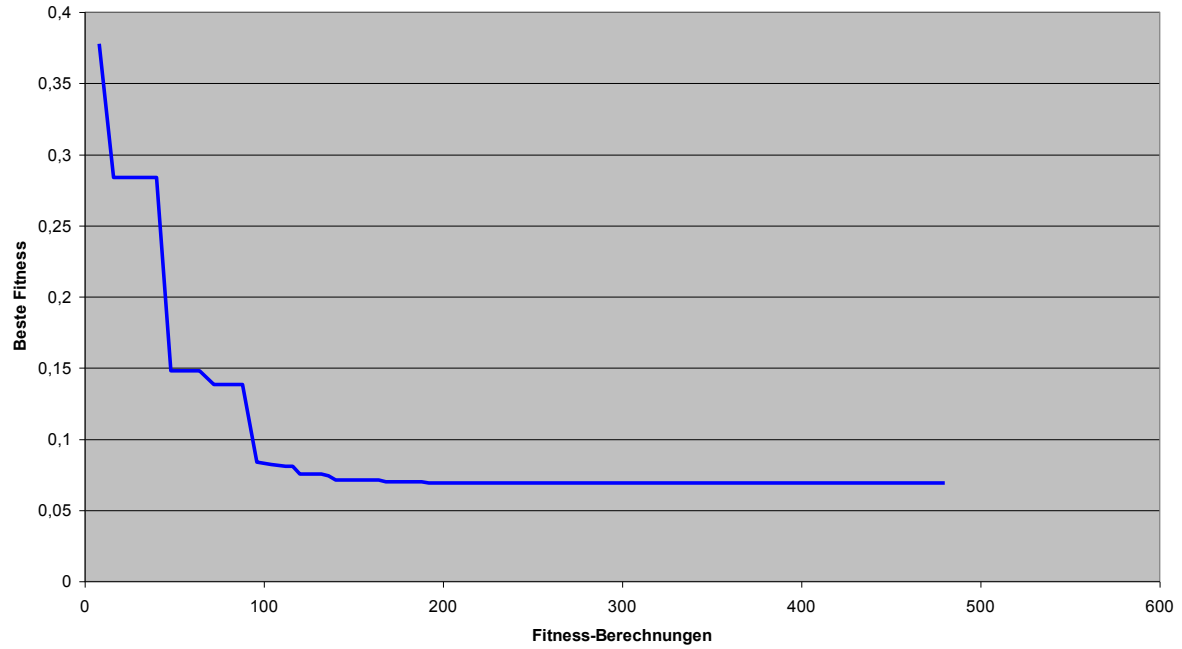
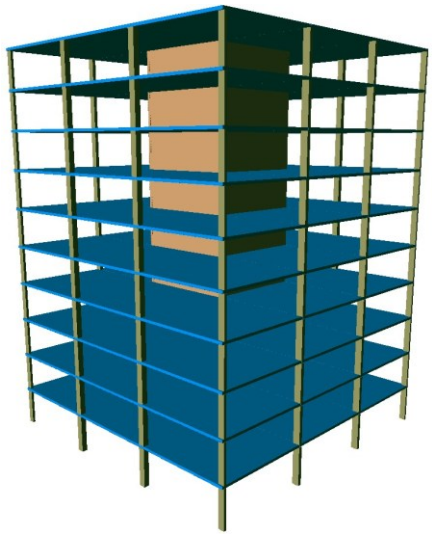
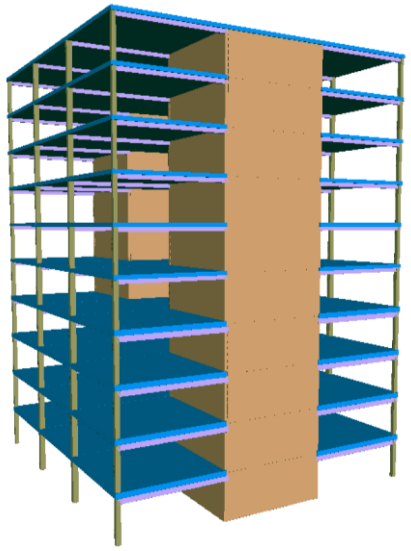


Abbildung 7-4: Optimierungsverlauf Experiment 2

Tabelle 7-1: Ergebnisse der Optimierung

Individuum 1 Generation 100	GEB[FELDER_X=3.00000000;FELDER_Y=3.00000000;RASTER_X=7.60000000;RASTER_Y=7.60000000;KERN[KFELD_X_PRZ=0.51000000;KFELD_Y_PRZ=0.06000000;KFELDER_X_PRZ=0.22000000;KFELDER_Y_PRZ=0.00000000;KMAX_STW=4.00000000;]KERN[KFELD_X_PRZ=0.54000000;KFELD_Y_PRZ=0.62000000;KFELDER_X_PRZ=0.09000000;KFELDER_Y_PRZ=0.40000000;KMAX_STW=13.00000000;]SWG[ANZ_STW=0;H_STW=3.10000000;OUTRIGGER=2.0;DS=FD;H_D=0.20000000;SSG[NAME=x-edge;DX=0.51000000;DY=0.42000000;]SSG[NAME=y-edge;DX=0.54000000;DY=0.28000000;]SSG[NAME=corner;DX=0.49000000;DY=0.20000000;]SSG[NAME=inside;DX=0.17000000;DY=0.26000000;]]]	 F = 0,055421
Individuum 2 Generation 100	GEB[FELDER_X=3.00000000;FELDER_Y=3.00000000;RASTER_X=7.60000000;RASTER_Y=7.60000000;KERN[KFELD_X_PRZ=0.61000000;KFELD_Y_PRZ=0.01000000;KFELDER_X_PRZ=0.12000000;KFELDER_Y_PRZ=0.02000000;KMAX_STW=4.00000000;]KERN[KFELD_X_PRZ=0.66000000;KFELD_Y_PRZ=0.70000000;KFELDER_X_PRZ=0.08000000;KFELDER_Y_PRZ=0.37000000;KMAX_STW=13.00000000;]SWG[ANZ_STW=5.00000000;H_STW=3.30000000;OUTRIGGER=2.0;DS=UZD;H_D=0.37000000;H_UZ=0.39000000;B_UZ=0.25000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.44000000;DY=0.46000000;]SSG[NAME=y-edge;DX=0.39000000;DY=0.37000000;]SSG[NAME=corner;DX=0.32000000;DY=0.28000000;]SSG[NAME=inside;DX=0.35000000;DY=0.25000000;]]SWG[ANZ_STW=0;H_STW=3.10000000;OUTRIGGER=2.0;DS=UZD;H_D=0.27000000;H_UZ=0.28000000;B_UZ=0.21000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.39000000;DY=0.40000000;]SSG[NAME=y-edge;DX=0.37000000;DY=0.29000000;]SSG[NAME=corner;DX=0.27000000;DY=0.28000000;]SSG[NAME=inside;DX=0.25000000;DY=0.28000000;]]]	 F = 0,615950

7.3 Experiment 3: Tragwerk mit 9 Stockwerken, Raster (4-6)*(4-6) Felder

In diesem Experiment wurden erneut die Optimierungsparameter sowie die Bildungsvorschrift aus Experiment 2 (Kap. 7.2) verwendet. Als einzige Veränderung an der Bildungsvorschrift wurde der Definitionsbereich des Grundrasters gesteigert (Listing 7-6). Durch das erweiterte Grundraster wird eine größere Variabilität der Ausbildung zusammengesetzter Querschnitte (Kerne) während des Optimierungsprozesses erreicht.

```
Felder:=[4;6;1]
```

Listing 7-6: Abweichung der Bildungsvorschrift Experiment 3

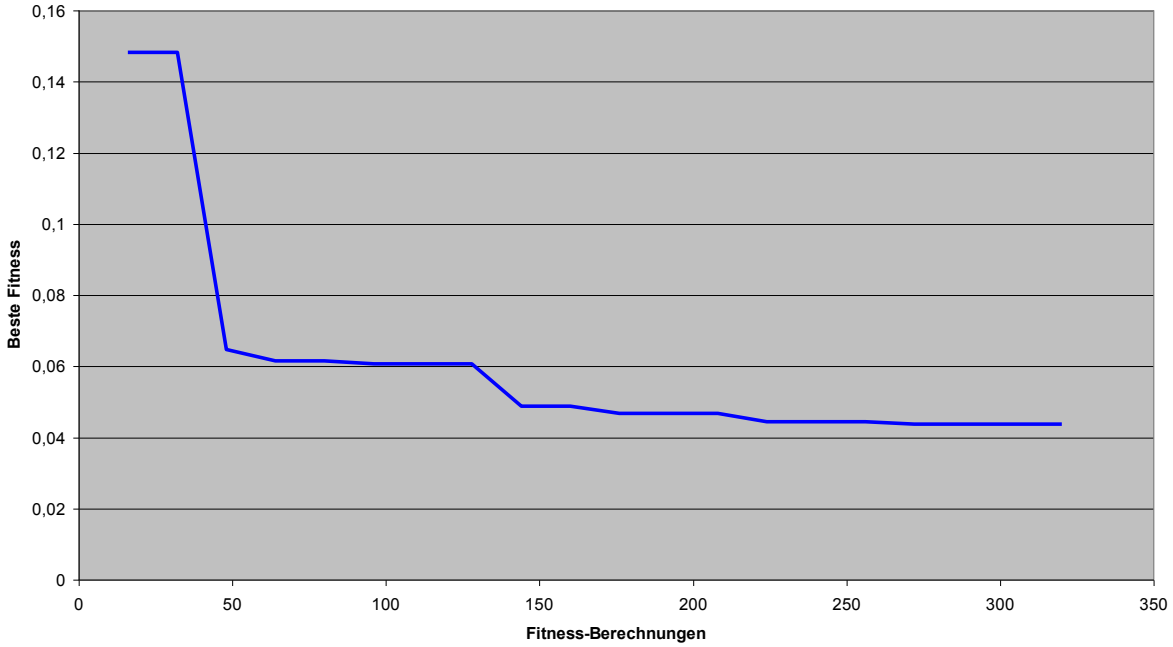
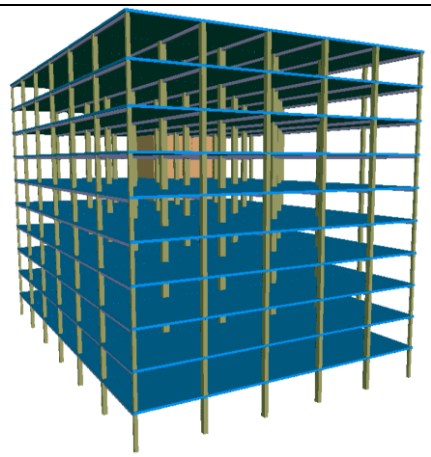
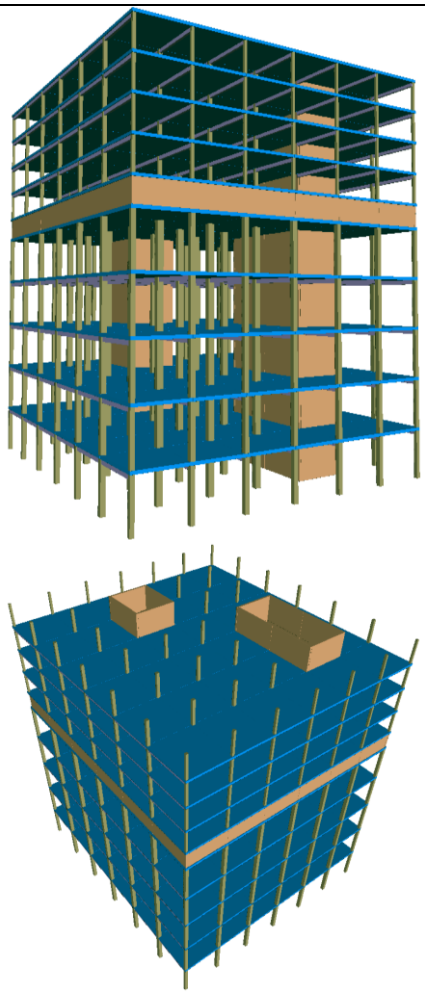


Abbildung 7-5: Optimierungsverlauf Experiment 3

Tabelle 7-2: Ergebnisse der Optimierung

Bestes Individuum Gen. 11	GEB[FELDER_X=5.00000000;FELDER_Y=6.00000000;RASTER_X=5.45000000;RASTER_Y=6.90000000;KERN[KFELD_X_PRZ=0.97000000;KFELD_Y_PRZ=0.93000000;KFELDER_X_PRZ=0.32000000;KFELDER_Y_PRZ=0.06000000;KMAX_STW=2.00000000;]KERN[KFELD_X_PRZ=0.61000000;KFELD_Y_PRZ=0.86000000;KFELDER_X_PRZ=0.30000000;KFELDER_Y_PRZ=0.16000000;KMAX_STW=7.00000000;]SWG[ANZ_STW=0;H_STW=3.00000000;OUTRIGGER=2.0;DS=UZD;H_D=0.20000000;H_UZ=0.26000000;B_UZ=0.21000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.35000000;DY=0.47000000;]SSG[NAME=y-edge;DX=0.37000000;DY=0.36000000;]SSG[NAME=corner;DX=0.28000000;DY=0.26000000;]SSG[NAME=inside;DX=0.53000000;DY=0.48000000;]]]	 F = 0,046827

Weiteres Individuum Gen. 8 (zur Veranschaulichung)	GEB[FELDER_X=6.00000000;FELDER_Y=6.00000000;RASTER_X=5.05000000;RASTER_Y=5.60000000;KERN[KFELD_X_PRZ=0.78000000;KFELD_Y_PRZ=0.27000000;KFELDER_X_PRZ=0.08000000;KFELDER_Y_PRZ=0.40000000;KMAX_STW=15.00000000;]KERN[KFELD_X_PRZ=0.47000000;KFELD_Y_PRZ=0.69000000;KFELDER_X_PRZ=0.30000000;KFELDER_Y_PRZ=0.21000000;KMAX_STW=19.00000000;]SWG[ANZ_STW=5.00000000;H_STW=4.90000000;OUTRIGGER=2.0;DS=UZD;H_D=0.30000000;H_UZ=0.40000000;B_UZ=0.28000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.42000000;DY=0.46000000;]SSG[NAME=y-edge;DX=0.46000000;DY=0.49000000;]SSG[NAME=corner;DX=0.39000000;DY=0.36000000;]SSG[NAME=inside;DX=0.58000000;DY=0.53000000;]]SWG[ANZ_STW=1;H_STW=2.40000000;OUTRIGGER=1.0;DS=FD;H_D=0.31000000;SSG[NAME=x-edge;DX=0.44000000;DY=0.37000000;]SSG[NAME=y-edge;DX=0.40000000;DY=0.41000000;]SSG[NAME=corner;DX=0.31000000;DY=0.35000000;]SSG[NAME=inside;DX=0.41000000;DY=0.59000000;]]SWG[ANZ_STW=6.00000000;H_STW=3.10000000;OUTRIGGER=2.0;DS=UZD;H_D=0.26000000;H_UZ=0.33000000;B_UZ=0.24000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.26000000;DY=0.27000000;]SSG[NAME=y-edge;DX=0.33000000;DY=0.31000000;]SSG[NAME=corner;DX=0.27000000;DY=0.25000000;]SSG[NAME=inside;DX=0.34000000;DY=0.30000000;]]SWG[ANZ_STW=1;H_STW=3.40000000;OUTRIGGER=1.0;DS=UZD;H_D=0.34000000;H_UZ=0.37000000;B_UZ=0.22000000;DIR_UZ=x;SSG[NAME=x-edge;DX=0.39000000;DY=0.25000000;]SSG[NAME=y-edge;DX=0.24000000;DY=0.22000000;]SSG[NAME=corner;DX=0.31000000;DY=0.60000000;]SSG[NAME=inside;DX=0.37000000;DY=0.43000000;]]SWG[ANZ_STW=1.00000000;H_STW=4.70000000;OUTRIGGER=2.0;DS=UZD;H_D=0.29000000;H_UZ=0.25000000;B_UZ=0.26000000;DIR_UZ=y;SSG[NAME=x-edge;DX=0.27000000;DY=0.31000000;]SSG[NAME=y-edge;DX=0.24000000;DY=0.25000000;]SSG[NAME=corner;DX=0.23000000;DY=0.22000000;]SSG[NAME=inside;DX=0.33000000;DY=0.31000000;]]SWG[ANZ_STW=0;H_STW=3.70000000;OUTRIGGER=2.0;DS=FD;H_D=0.40000000;SSG[NAME=x-edge;DX=0.26000000;DY=0.27000000;]SSG[NAME=y-edge;DX=0.22000000;DY=0.25000000;]SSG[NAME=corner;DX=0.22000000;DY=0.21000000;]SSG[NAME=inside;DX=0.32000000;DY=0.33000000;]]]	 F = 0,176766

7.4 Fazit

Die in den Ergebnissen der experimentellen Versuchsreihen dargestellten Optimierungsverläufe sowie die exemplarisch gezeigten Individuen demonstrieren deutlich die Eignung der entwickelten Konzepte zur Optimierung komplexer Ingenieurprobleme am Beispiel von Tragstrukturen des Hochbaus.

Hierbei gilt insbesondere zu beachten, dass das Ziel der vorliegenden Arbeit nicht in der Erstellung optimierter Tragstrukturen lag, sondern in der Entwicklung anpassbarer Konzepte, die eine eigenständige Optimierung von Bauwerken unter Beachtung gegebener Randbedingungen ermöglichen. Somit standen nicht die Bauwerke selbst, sondern die Evaluation des Optimierungssystems im Mittelpunkt der Untersuchungen. Auch die Praxisrelevanz der beispielhaft entwickelten Bauwerke oder die tatsächlich erreichbare Steigerung von Wirtschaftlichkeit und weiterer Parameter ist hier von untergeordneter Bedeutung. Die durchgeführten Versuche wurden aufgrund der Komplexität des Berechnungsvorgangs auf relativ kleine Bauwerke angewendet, die jedoch zur qualitativen Evaluation der entwickelten Konzepte ausreichend sind. Die Übertragung der Konzepte auf komplexere Problemstellungen und Bauwerke ist ohne weiteres möglich.

Die Ergebnisse der Experimente zeigen deutlich die Fähigkeit des entwickelten Systems, anhand der vorgegebenen Randbedingungen, Optimierungskriterien und wissensbasiert formalisierten Berechnungsvorschriften eine signifikante Verbesserung bewertbarer Größen eines gegebenen Optimierungsproblems zu erreichen.

8 Zusammenfassung

Im Rahmen der vorliegenden Arbeit wurde ein umfassendes technisches Konzept zur softwarebasierten Optimierung dreidimensionaler Tragstrukturen erarbeitet. Hierzu kommt das Prinzip der genetischen Programmierung zur Anwendung. Anstelle der Definition von Funktionen und Symbolen wurde ein grammatik-basierter Ansatz auf Grundlage der Backus-Naur Form zur Notation der Bildungsvorschrift für die genetische Programmierung gewählt, der um wesentliche Elemente und Operatoren zur Repräsentation ingenieurtechnischer Problemstellungen erweitert wurde. Die Bildungsvorschrift der Optimierungskomponente ist durch den Anwender ohne Programmierkenntnisse erweiterbar, wodurch ein hoher Grad an Flexibilität erreicht wird.

Weiterhin wurden Methoden zur wissensbasierten Transformation genotypischer Entwurfs-Informationen in bewertbare, phänotypische Ausprägungen entwickelt. Der wissensbasierte Ansatz ermöglicht die Anpassung und Erweiterung der zur Erstellung des phänotypischen Tragwerksmodells notwendigen Entwurfsregeln. In Verbindung mit der anpassbaren Bildungsvorschrift der Optimierungskomponente erlaubt dieses Konzept die freie Definition unterschiedlicher Entwurfselemente durch den Anwender sowie deren direkte Anwendung im Rahmen einer Bauwerksoptimierung.

Die Bewertung der „Fitness“ eines Bauwerks erfolgt ebenfalls durch einen wissensbasierten Ansatz. Dies umfasst die Bestimmung maßgebender Bauwerkseigenschaften, die Interpretation der phänotypischen Informationen für die Bemessung und den Nachweis von Strukturelementen sowie die ganzheitliche Bewertung des gesamten Bauwerks mittels wissensbasierter Bewertungsregeln. Der wissensbasierte Ansatz ermöglicht eine freie Definition beliebiger Optimierungskriterien durch den Anwender.

Die entwickelten Modelle und Konzepte wurden in mehreren Software-Applikationen prototypisch implementiert. Die Hauptanwendung beinhaltet Komponenten zur Visualisierung und Optimierung von Tragstrukturen. Zwei weitere Anwendungen dienen der Erstellung und Pflege von Wissensbanken sowie der für den Optimierungsprozess notwendigen Bildungsvorschrift. Die Optimierungs-Plattform bietet Funktionalitäten zur Verteilung der Rechenlast auf mehrere Rechner über das Internet.

Durch Experimente konnte gezeigt werden, dass die entwickelten Konzepte zur grundsätzlichen Optimierung von Tragsystemen gut geeignet sind. Der hohe Grad an Modularisierung und die klare Trennung von Softwarekomponenten, Bildungsvorschrift, Entwurfs- und Bewertungswissen erlauben die teilweise oder vollständige Übertragung der entwickelten Ansätze auf weitere Optimierungsprobleme. Mit geringfügigen Anpassungen ist auch die Anwendung des Systems auf Probleme aus völlig artfremden Disziplinen möglich.

Ein wesentlicher Beitrag der vorliegenden Arbeit liegt in der inhaltlichen Entkopplung komplexer Optimierungs-, Modellierungs- und Bewertungsverfahren von deren softwaretechnischen Implementierung. Die dadurch erreichte Anpassbarkeit sämtlicher Stufen des Optimierungsprozesses erschließt dem Anwender ein wesentlich breiteres Einsatzspektrum, als dies durch übliche, auf spezifische Problemstellungen zugeschnittene Optimierungssysteme möglich ist.

8.1 Weiterer Forschungsbedarf

8.1.1 Erweiterung von Bildungsvorschrift und Wissensbanken

Das entwickelte Softwarekonzept bildet die Grundlage einer Optimierungsplattform. Zur Optimierung von Tragstrukturen sind zusätzlich entsprechende Bildungsvorschriften und Wissensbanken für Design und Bewertung erforderlich. Im Rahmen der vorliegenden Arbeit wurden beispielhafte Bildungsvorschriften und Wissensbanken erstellt, die zur Verifikation der entwickelten Softwarekonzepte ausreichend sind.

Zur weiteren Evaluation der praktischen Anwendbarkeit der entwickelten Konzepte ist die Durchführung umfangreicher Experimente notwendig. Ziel der Experimente sollte, neben der Verbesserung des Optimierungsverhaltens, die automatisierte Entwicklung innovativer Tragstrukturen sein, die einen signifikanten Mehrwert zu traditionellen Gebäudeentwürfen darstellen.

8.1.2 Verbesserung der Fitness-Evaluation

Die in den vorgestellten Experimenten durchgeführte Fitness-Evaluation durchläuft einen aufwändigen Prozess. Wesentlicher Teil dieses Prozesses ist die detaillierte Finite Elemente-Berechnung. Durch die dreidimensionale Modellierung steigt die Anzahl der finiten Elemente bereits mit der Erweiterung des Bauwerksmodells in einer einzelnen Achse linear an. Eine wesentliche Steigerung der Gebäudehöhe erfordert jedoch auch die Vergrößerung des Grundrisses, wodurch mit einer kubischen Zunahme der Anzahl der finiten Elemente gerechnet werden muss. Zudem nimmt der Rechenaufwand mit steigender Elementanzahl überproportional zu.

Zur effizienten Berechnung von Hochhäusern sind daher wesentliche Vereinfachungen der FE-Modellierung erforderlich, wie beispielsweise die Verringerung der Elementanzahl je Strukturelement, oder die Zusammenfassung von ähnlich belasteten Strukturelementen zu Berechnungsgruppen. Diese Vereinfachungen dürfen jedoch nicht zu unerwünschten Fehlbewertungen der Bauwerke führen, so dass sich die Abweichung im Vergleich zu einem hinreichend detailliert berechneten Bauwerk insgesamt unterhalb der für die Bewertung relevanten Toleranzgrenzen bewegen muss. Eine detaillierte Untersuchung dieses Aspektes war nicht Teil der vorliegenden Arbeit

8.1.3 Erweiterung der Grundrissgestaltung

Neben den grundlegenden Erweiterungen der Bildungsvorschrift sowie der verwendeten Wissensbanken ist zur Ausbildung innovativer Tragstrukturen auch die Verarbeitung komplexer Grundrissformen erforderlich. Hierfür wäre eine Kombination vorgegebener Grundformen zu komplexeren zusammengesetzten Grundrissen möglich. Eine große Herausforderung ist in diesem Rahmen die wissensbasierte Interpretierbarkeit komplexer, in beliebigen Formen und Richtungen vernetzter Topologiemodelle.

8.1.4 Berücksichtigung von Tiefgeschossen und Gründung

Die Beispiele der vorliegenden Arbeit beschränken sich auf die oberen Geschosse und berücksichtigen nicht die Gründung. Tiefgeschosse und Gründung haben jedoch einen wesentlichen Einfluss auf die Gesamtkosten sowie die Nutzbarkeit eines Bauwerks. Die Wahl und Dimensionierung des Gründungssystems in Abstimmung mit dem Tragwerk bietet ein enormes Optimierungspotential.

8.1.5 Multikriterielle Optimierung

Die vorgestellten Anwendungsbeispiele beinhalten mehrere Bewertungsfunktionen, die zu einem einzelnen Fitnesswert zusammengefasst werden. In vielen Fällen ist die Rückführung mehrerer Kriterien auf ein Gesamtkriterium nicht gewünscht oder sogar kontraproduktiv. Gerade in Fällen nicht eindeutiger und vollumfänglich formalisierbarer Bewertungskriterien, wie im Bereich automatisierter Designs, kann es eher zielführend sein, mehrere Kriterien gleichermaßen und unabhängig voneinander in die Optimierung mit einzubeziehen. Ein Ansatz hierfür ist die Anwendung multikriterieller Optimierung.

8.1.6 Selbstadaptierende Optimierungsparameter

Die Wahl der Optimierungsparameter hat enormen Einfluss auf das Optimierungsverhalten und die Effizienz eines Optimierungssystems. In Kap. 4.2.8 wurde auf die generationsabhängige Adaption der Optimierungsparameter eingegangen. Die manuelle Wahl der Parameter und Generationsgrenzen gestaltet sich relativ schwierig. Im Laufe des Optimierungsprozesses entstehen häufig Situationen, die eine manuelle Anpassung der Parameter erforderlich machen.

Eine Erweiterung der Optimierungskomponente um selbstadaptierende Parameter könnte zu einer wesentlichen Verbesserung des Optimierungsverhaltens führen. Die Bestimmung der Optimierungsparameter würde vereinfacht. Suboptimale Parametereinstellungen wären unkritisch, da diese durch Selbstadaption in wenigen Generationen automatisch ausgeglichen werden könnten.

8.1.7 Fuzzy-Logik

Die Formalisierung der Bewertungsfunktionen erfordert oftmals die Hinzunahme "unscharfer" Parameter, die in geeigneter Weise mathematisch erfasst und verarbeitet werden müssen. Die numerische Quantifizierung zu frühen Zeitpunkten der Berechnung birgt die Gefahr von Informationsverlusten, die sich bei weiterer Verarbeitung fortpflanzen und verstärken. Durch den Einsatz von Fuzzy-Logik können unscharfe Werte adäquat und frei von Informationsverlusten repräsentiert und verarbeitet werden.

8.1.8 Anwendung im Entwurf von Brücken

Die Grundprinzipien der vorliegenden Arbeit lassen sich sehr gut auf Probleme weiterer Ingenieurbauwerke erweitern, wie beispielsweise dem Entwurf von Brücken. Mögliche Parameter der Optimierung von Brückenbauwerken sind beispielsweise:

- Festlegung der Bauweise (Beton, Stahl, Stahlverbund)
- Wahl und Dimensionierung der Querschnittsform (Hohlkasten-, Plattenbalkenquerschnitt)
- Optimierung des globalen Längsprofils (Spannweiten, Widerlager, Stützen, Gründung)
- Optimierung des lokalen Längsprofils einzelner Felder

9 Anhang

9.1 Anhang A: Zusammenstellung der Design-Wissensbank

Id	Type	Identifier	Name	Product-model	Content
0	Formula	ANZ_STW	[ANZ_STW_TMP]*2		[ANZ_STW_TMP]*2
2	From Product-model (Derived)	Edge_North	Edge_North	Mesh_Node	Edge_North Connected Edge, highest available Level
3	From Product-model (Derived)	Edge_South	Edge_South	Mesh_Node	Edge_South Connected Edge, highest available Level
4	From Product-model (Derived)	Edge_West	Edge_West	Mesh_Node	Edge_West Connected Edge, highest available Level
5	From Product-model (Derived)	Edge_East	Edge_East	Mesh_Node	Edge_East Connected Edge, highest available Level
6	From Product-model (Derived)	Edge_Top	Edge_Top	Mesh_Node	Edge_Top Connected Edge, highest available Level
7	From Product-model (Derived)	Edge_Bottom	Edge_Bottom	Mesh_Node	Edge_Bottom Connected Edge, highest available Level
8	From Product-model (Derived)	Node_Left	Node_1	Mesh_Edge	Node_1 Node 1 (Min x/y/z, dep. on Direction)
9	From Product-model (Derived)	Node_Right	Node_2	Mesh_Edge	Node_2 Node 2 (Max x/y/z, dep. on Direction)
10	Formula	Node_North	[Edge_North.Node_Right]	Mesh_Node	[Edge_North.Node_Right]
11	Formula	Edge_NorthEast	[Node_North.Edge_East]	Mesh_Node	[Node_North.Edge_East]
12	Formula	Edge_West	[Node Left.Edge North]	Mesh_Edge	[Node_Left.Edge_North]
13	Formula	Edge_East	[Node Right.Edge North]	Mesh_Edge	[Node_Right.Edge_North]
14	Formula	Edge_North	[Edge_West.Edge_NorthEast]	Mesh_Edge	[Edge_West.Edge_NorthEast]
15	Formula	Edge_NorthEast	[Node_Right.Edge_East]	Mesh_Edge	[Node_Right.Edge_East]
16	Design selection		Slab	Mesh_Edge	Requirement rule: ([Kern_N]=0 OR [Floor_Above]<0) AND [Floor_Below]>=0
17	From Product-model (Basic)	x	POS_X	Mesh_Node	POS_X delivers the absolute x-position of the element
18	From Product-model (Basic)	y	POS_Y	Mesh_Node	POS_Y delivers the absolute y-position of the element
19	From Product-model (Basic)	z	POS_Z	Mesh_Node	POS_Z delivers the absolute z-position of the element
20	Formula	slab_dx	[Node_Right.x]-[Node_Left.x]+[Slab_Offset_West]+[Slab_Offset_East]	Mesh_Edge	[Node_Right.x]-[Node_Left.x]+[Slab_Offset_West]+[Slab_Offset_East]

21	Formula	NorthY	[Node_Right.y]	Mesh_Edge	[Node_Right.y]
22	Formula	slab_dy	[Edge_West.NorthY]-[Node_Left.y]+[Slab_Offset_North]+[Slab_Offset_South]	Mesh_Edge	[Edge_West.NorthY]-[Node_Left.y]+[Slab_Offset_North]+[Slab_Offset_South]
24	Design selection		Column	Mesh_Edge	Requirement rule: [Kern_E]=0 AND [Kern_N]=0 AND [Kern_S]=0 AND [Kern_W]=0 AND [DropColumnOnOutrigger]=0
25	Constant	Col_dx	0.3000	Mesh_Edge	Value=0.3000
26	Constant	Col_dy	0.3000	Mesh_Edge	Value=0.3000
27	Formula	Col_dz	abs([Node_Right.z]-[Node_Left.z])-[Col_Ofs_1]-[Col_Ofs_2]	Mesh_Edge	abs([Node_Right.z]-[Node_Left.z])-[Col_Ofs_1]-[Col_Ofs_2]
28	Formula	H_D	[SWG.H_D]	Floor	[SWG.H_D]
29	From Product-model (Basic)	SWG	SWG	Floor	SWG delivers the floor optimisation group
30	From Product-model (Derived)	Floor_Above	FLOOR_ABOVE	Floor	FLOOR_ABOVE delivers the above floor
31	Formula	Col_Ofs_2	[H_D]/2.0	Floor	[H_D]/2.0
32	Formula	Col_Ofs_1	[H_D]/2.0	Floor	[H_D]/2.0
33	From Product-model (Derived)	Kern_N	Kern_N	Mesh_Edge	Kern_N Kern in this direction
34	From Product-model (Derived)	Kern_S	Kern_S	Mesh_Edge	Kern_S Kern in this direction
35	From Product-model (Derived)	Kern_W	Kern_W	Mesh_Edge	Kern_W Kern in this direction
36	From Product-model (Derived)	Kern_E	Kern_E	Mesh_Edge	Kern_E Kern in this direction
37	Design selection		Wall_X_unten	Mesh_Edge	Requirement rule: ([Kern_N]=1 AND [Kern_S]=0) OR ([OUTRIGGER]=1 AND [Edge_Type]='x-edge')
38	Formula	Edge_WallLeft	[Node_Left.Edge_Top]	Mesh_Edge	[Node_Left.Edge_Top]
39	Formula	Edge_WallRight	[Node_Right.Edge_Top]	Mesh_Edge	[Node_Right.Edge_Top]
40	Formula	Edge_TopWallTopEast	[Node_Right.Edge_East]	Mesh_Edge	[Node_Right.Edge_East]
41	Formula	Edge_TopWallTopNorth	[Node_Right.Edge_North]	Mesh_Edge	[Node_Right.Edge_North]
42	Formula	Edge_WallTopEast	[Edge_WallLeft.Edge_TopWallTopEast]	Mesh_Edge	[Edge_WallLeft.Edge_TopWallTopEast]
43	Formula	Edge_WallTopNorth	[Edge_WallLeft.Edge_TopWallTopNorth]	Mesh_Edge	[Edge_WallLeft.Edge_TopWallTopNorth]
44	Design selection		Wall_Y_links	Mesh_Edge	Requirement rule: ([Kern_W]=0 AND [Kern_E]=1) OR ([OUTRIGGER]=1 AND [Edge_Type]='y-edge')
45	Formula	Wall_dx_X	[Node_Right.x]-[Node_Left.x]	Mesh_Edge	[Node_Right.x]-[Node_Left.x]
46	Formula	Wall_dx_Y	[Node_Right.y]-[Node_Left.y]	Mesh_Edge	[Node_Right.y]-[Node_Left.y]
47	Formula	Wall_dy	.24	Mesh_Edge	.24
48	Formula	Wall_dz	[Edge_WallLeft.Edge_TopWallTopZ]-[Node_Left.z]	Mesh_Edge	[Edge_WallLeft.Edge_TopWallTopZ]-[Node_Left.z]
49	Formula	Edge_TopWallTopZ	[Node_Right.z]	Mesh_Edge	[Node_Right.z]
50	Formula	Wall_Offset_l	[Wall_dy]/2	Mesh_Edge	[Wall_dy]/2
51	Formula	Wall_Offset_r	[Wall_dy]/(-2)	Mesh_Edge	[Wall_dy]/(-2)

54	Design selection		Wall_X_oben	Mesh_Edge	Requirement rule: ([Kern_N]=0 AND [Kern_S]=1)
55	Design selection		Wall_Y_rechts	Mesh_Edge	Requirement rule: ([Kern_W]=1 AND [Kern_E]=0)
58	Formula	Max_Col_dx	[SWG.Max_Col_DX]	Mesh_Edge	[SWG.Max_Col_DX]
59	Formula	Max_Col_dy	[SWG.Max_Col_DY]	Mesh_Edge	[SWG.Max_Col_DY]
60	Formula	Slab_Offset_West	[Max_Col_dx]/2	Mesh_Edge	[Max_Col_dx]/2
61	Formula	Slab_Offset_East	[Max_Col_dx]/2	Mesh_Edge	[Max_Col_dx]/2
62	Formula	Slab_Offset_North	[Max_Col_dy]/2	Mesh_Edge	[Max_Col_dy]/2
63	Formula	Slab_Offset_South	[Max_Col_dy]/2	Mesh_Edge	[Max_Col_dy]/2
64	Constant	Slab_Offset_West	0	Mesh_Edge	Value=0.0000
65	Constant	Slab_Offset_East	0	Mesh_Edge	Value=0.0000
66	Constant	Slab_Offset_North	0	Mesh_Edge	Value=0.0000
67	Constant	Slab_Offset_South	0	Mesh_Edge	Value=0.0000
68	Rule-set		Slab_Offset_West	Mesh_Edge	(1) if [Node_Left.Edge_West]<0, then use Phenotype1_60 (2) else use Phenotype1_64
69	Rule-set		Slab_Offset_East	Mesh_Edge	(1) if [Node_Right.Edge_East]<0, then use Phenotype1_61 (2) else use Phenotype1_65
70	Rule-set		Slab_Offset_North	Mesh_Edge	(1) if [Node_NorthWest.Edge_North]<0 AND [Floor_Above]<0 AND 1>2, then use Phenotype1_73 (2) else if [Node_NorthWest.Edge_North]<0 , then use Phenotype1_62 (3) else use Phenotype1_66
71	Rule-set		Slab_Offset_South	Mesh_Edge	(1) if [Node_Right.Edge_South]<0, then use Phenotype1_63 (2) else use Phenotype1_67
72	Formula	Node_NorthWest	[Edge_West.Node_Right]	Mesh_Edge	[Edge_West.Node_Right]
73	Formula	Slab_Offset_North	[Max_Col_dy]/2+1	Mesh_Edge	[Max_Col_dy]/2+1
74	Design selection		Beam_Main_X	Mesh_Edge	Requirement rule: ([DS]=('UZD') OR [DS]=('RD')) AND [DIR_UZ]=('x') AND [Beam_EdgeDownEast.Kern_N]=0 AND [Beam_EdgeDownEast.Kern_S]=0 AND NOT [Floor Below.OUTRIGGER]=1
75	Assign-ment	DS	Deckensystem	Floor	[SWG.DS]
76	Formula	H_UZ	[SWG.H_UZ]	Floor	[SWG.H_UZ]
77	Formula	B_UZ	[SWG.B_UZ]	Floor	[SWG.B_UZ]
78	Formula	Beam_Offset_S	[H_UZ]/2+[H_D]/2	Mesh_Edge	[H_UZ]/2+[H_D]/2
79	Assign-ment	DIR_UZ	Richtung	Mesh_Edge	[SWG.DIR_UZ]
80	Design selection		Beam_Main_Y	Mesh_Edge	Requirement rule: ([DS]=('UZD') OR [DS]=('RD')) AND [DIR_UZ]='y' AND [Beam_EdgeDownNorth.Kern_W]=0 AND [Beam_EdgeDownNorth.Kern_E]=0 AND NOT [Floor Below.OUTRIGGER]=1
81	From Product-model (Derived)	Floor_Below	FLOOR_BELOW	Floor	FLOOR_BELOW delivers the below floor

82	Formula	Beam_Edge_Down	[Node_Left.Edge_Bottom]	Mesh_Edge	[Node_Left.Edge_Bottom]
83	Formula	Beam_Edge_East	[Node_Left.Edge_East]	Mesh_Edge	[Node_Left.Edge_East]
84	Formula	Beam_EdgeDownEast	[Beam_Edge_Down.Beam_Edge_East]	Mesh_Edge	[Beam_Edge_Down.Beam_Edge_East]
85	Formula	Beam_Edge_North	[Node_Left.Edge_North]	Mesh_Edge	[Node_Left.Edge_North]
86	Formula	Beam_EdgeDownNorth	[Beam_Edge_Down.Beam_Edge_North]	Mesh_Edge	[Beam_Edge_Down.Beam_Edge_North]
87	From Product-model (Derived)	Edge_Type	Type	Mesh_Edge	Type Type of edge
88	From Product-model (Basic)	SSG_x_edge	SSG		
89	From Product-model (Basic)	SSG_y_edge	SSG		
90	From Product-model (Basic)	SSG_inside	SSG		
91	From Product-model (Basic)	SSG_corner	SSG		
92	Rule-set		Ssg	Mesh_Edge	(1) if [Edge_Type]='x-edge' , then use Phenotype1_97 (2) else if [Edge_Type]='y-edge' , then use Phenotype1_98 (3) else if [Edge_Type]='corner' , then use Phenotype1_95 (4) else if [Edge_Type]='inside' , then use Phenotype1_96
93	Formula	Col_dx	[SSG.DX]	Mesh_Edge	[SSG.DX]
94	Formula	Col_dy	[SSG.DY]	Mesh_Edge	[SSG.DY]
95	Formula	SSG	[SWG.SSG_corner]	Mesh_Edge	[SWG.SSG_corner]
96	Formula	SSG	[SWG.SSG_inside]	Mesh_Edge	[SWG.SSG_inside]
97	Formula	SSG	[SWG.SSG_x_edge]	Mesh_Edge	[SWG.SSG_x_edge]
98	Formula	SSG	[SWG.SSG_y_edge]	Mesh_Edge	[SWG.SSG_y_edge]

9.2 Anhang B: Zusammenstellung der Bewertungs-Wissensbank

Id	Type	Identifier	Name	Product-model	Content
99	Formula	Max_Col_DX	max(max([SSG_corner.DX];[SSG_inside.DX]);max([SSG_x_edge.DX];[SSG_y_edge.DX]))	Column	max(max([SSG_corner.DX];[SSG_inside.DX]);max([SSG_x_edge.DX];[SSG_y_edge.DX]))
100	Formula	Max_Col_DY	max(max([SSG_corner.DY];[SSG_inside.DY]);max([SSG_x_edge.DY];[SSG_y_edge.DY]))	Column	max(max([SSG_corner.DY];[SSG_inside.DY]);max([SSG_x_edge.DY];[SSG_y_edge.DY]))
101	Constant	Max_Col_DY	0.4000	Column	Value=0.4000
102	Constant	Max_Col_DX	0.4000	Column	Value=0.4000
103	Komplex table	E_Modul;Nue;Gamma;	Betonkennwerte	Element	complex content
104	Constant	Betonfes-tigkeits-klasse	C30/37	Element	Value=C30/37
105	Formula	Gamma_Dyn	[Gamma]	Element	[Gamma]
106	Constant	Gamma_Dyn	0	Column	Value=0.0000
107	Constant	AN-ZAHL_STW_GES	11	Building Template	Value=11.0000
108	Formula	ANZ_STW	[ANZAHL_STW_GES]-[Floor_Count]	Building Template	[ANZAHL_STW_GES]-[Floor_Count]
109	From Product-model (Derived)	To-tal_Heigth	TOTAL_HEIGTH	Building Template	
110	From Product-model (Derived)	Floor_Count	FLOOR_COUNT	Building	
111	Formula	DX	[FELDER_X]*[RASTER_X]	Building Template	[FELDER_X]*[RASTER_X]
112	Formula	DY	[FELDER_Y]*[RASTER_Y]	Building Template	[FELDER_Y]*[RASTER_Y]
113	Constant	FELDER_X	1	Building Template	Value=1.0000
114	Constant	FELDER_Y	1	Building Template	Value=1.0000
115	Constant	RASTER_X	1	Building Template	Value=1.0000
116	Constant	RASTER_Y	1	Building Template	Value=1.0000
117	Constant	FELD_X_PRZ	1	Kern Temp-late	Value=1.0000
118	Constant	FELD_Y_PRZ	1	Kern Temp-late	Value=1.0000
119	Constant	FEL-DER_X_PRZ	1	Kern Temp-late	Value=1.0000
120	Constant	FEL-DER_Y_PRZ	1	Kern Temp-late	Value=1.0000

Id	Type	Identifier	Name	Product-model	Content
121	Formula	KFELD_X	max(roundup(([BW_FELDER_X]-[KFEL-DER_X]+1)*[KFELD_X_PRZ]);1)	Kern Template	max(roundup(([BW_FELDER_X]-[KFEL-DER_X]+1)*[KFELD_X_PRZ]);1)
122	Formula	KFELDER_X	max(round([KFELDER_X_PRZ]*[BW_FELDER_X]);1)	Kern Template	max(round([KFELDER_X_PRZ]*[BW_FELDER_X]);1)
125	Formula	KFELDER_Y	max(round([KFELDER_Y_PRZ]*[BW_FELDER_Y]);1)	Kern Template	max(round([KFELDER_Y_PRZ]*[BW_FELDER_Y]);1)
126	Formula	KFELD_Y	max(roundup(([BW_FELDER_Y]-[KFEL-DER_Y]+1)*[KFELD_Y_PRZ]);1)	Kern Template	max(roundup(([BW_FELDER_Y]-[KFEL-DER_Y]+1)*[KFELD_Y_PRZ]);1)
127	Formula	BW_FELDER_X	[FELDER_X]	Building Template	[FELDER_X]
128	Formula	BW_FELDER_Y	[FELDER_Y]	Building Template	[FELDER_Y]
130	Constant	Erdbebenzone	1	Building	Value=1
131	Constant	Untergrund-verhaeltnis	A-R	Building	Value=A-R
132	Constant	Bedeutungs-kategorie	1	Building	Value=1
133	Constant	Windzone	1	Building	Value=1
134	Constant	Gelaendeka-tegorie	1	Building	Value=1
135	Komplex table	S_h;T_B_h;T_C_h;T_D_h;	Untergrundverhaeltnis	Building	complex content
136	Komplex table	a_g;	Erdbebenzone	Building	complex content
137	Komplex table	gamma_I;	Bedeutungskategorie	Building	complex content
138	Komplex table	f_v_m;e_v_m;f_I;e_I;f_v_b;e_v_b;epsi-lon;z min;	Gelaendekategorie	Building	complex content
139	Komplex table	v_ref;	Windzone	Building	complex content
140	Constant	beta_0	2,5	Building	Value=2.5000
142	Constant	daempf	0,05	Building	Value=0.0500
143	From Product-model (Basic)	Opt_Template	Template	Building	Template Optimization Template
144	Formula	Dim_X	[Opt_Template.FELDER_X]*[Opt_Template.RASTER_X]	Building	[Opt_Template.FELDER_X]*[Opt_Template.RASTER_X]

Id	Type	Identifier	Name	Product-model	Content
145	Formula	Dim_Y	[Opt_Template.FELDER_Y]*[Opt_Template.RASTER_Y]	Building	[Opt_Template.FELDER_Y]*[Opt_Template.RASTER_Y]
148	Constant	Phi_unten	0,7	Building	Value=0.7000
149	Constant	Phi_oben	1	Building	Value=1.0000
150	Formula	v_ref_0	[v_ref]	Building	[v_ref]
151	Constant	Fitness1	0.8000	Building	Value=0.8000
152	Constant	Fitness2	0.6000	Building	Value=0.6000
153	Formula	Fitness1	rndn(1000)/1000	Building	rndn(1000)/1000
154	Formula	Fitness2	[Fitness1]	Building	[Fitness1]
155	Komplex table	q_k;	Verkehrslasten	Floor	complex content
156	Constant	Nutzungsart	Büro	Floor	Value=Büro
157	Formula	OUTRIGGER	[SWG.OUTRIGGER]	Floor	[SWG.OUTRIGGER]
159	Constant	DropColumnOnOutrigger	True		Value=1.0000
160	Constant	DropColumnOnOutrigger	False		Value=0.0000
161	Rule-set		DropColumnOnOutrigger		(1) if [NAME]='inside', then use Phenotype1_160 (2) else use Phenotype1_159
162	Constant	DropColumnOnOutrigger	True	Mesh_Edge	Value=1.0000
163	Constant	DropColumnOnOutrigger	False	Mesh_Edge	Value=0.0000
164	Rule-set		DropColumnOnOutrigger	Mesh_Edge	(1) if [SWG.OUTRIGGER]=1 AND [SSG.DropColumnOnOutrigger]=1, then use Phenotype1_162 (2) else use Phenotype1_163
166	Formula	Volumen	[dx]*[dy]*[dz]	Element	[dx]*[dy]*[dz]
167	From Product-model (Basic)	dx	EXT_X	Element	EXT_X delivers the total x-extent of the element
168	From Product-model (Basic)	dy	EXT_Y	Element	EXT_Y delivers the total y-extent of the element
169	From Product-model (Basic)	dz	EXT_Z	Element	EXT_Z delivers the total z-extent of the element
170	Komplex table	q;	Duktilitätsbeiwert	Building	complex content
171	Constant	Duktilitaetsklasse	Duktilitätsklasse	Building	Value=1.0000

Id	Type	Identifier	Name	Product-model	Content
172	From Product-model (Derived)	Dim_Z	TOTAL_HEIGTH	Building	TOTAL_HEIGTH Total building heigth
173	Formula	Z_Eff	0,6*[Dim_Z]	Building	0,6*[Dim_Z]
174	Formula	Z_Eff	[z_min]	Building	[z_min]
175	Rule-set		Z_Eff	Building	(1) if 0,6*[Dim_Z]>[z_min], then use Phenotype1_173 (2) else use Phenotype1_174
176	Formula	N_1x	[Eigenfre-quenz]*[L_I_zeff]/[v_m_zeff]	Building	[Eigenfre-quenz]*[L_I_zeff]/[v_m_zeff]
177	From Product-model (Basic)	Eigenfre-quenz	EIGENFRQ	Building	EIGENFRQ First eigen frequency
178	Formula	L_I_zeff	300*([Z_Eff]/300)^[epsilon]	Building	300*([Z_Eff]/300)^[epsilon]
179	Formula	L_I_zeff	300*([z_min]/300)^[epsilon]	Building	300*([z_min]/300)^[epsilon]
180	Formula	L_I_zeff	300	Building	300
181	Rule-set		L_I_zeff	Building	(1) if [z_min]<=[Z_Eff] AND [Z_Eff]<=300, then use Phenotype1_178 (2) else if [Z_Eff]<[z_min], then use Phenotype1_179 (3) else use Phenotype1_180
182	Formula	v_m_zeff	[f_v_m]*[v_ref_0]*([Z_Eff]*10)^[e_v_m]	Building	[f_v_m]*[v_ref_0]*([Z_Eff]*10)^[e_v_m]
183	Formula	eta_h	4,6*[N_1x]*[Dim_Z]/[L_I_zeff]	Building	4,6*[N_1x]*[Dim_Z]/[L_I_zeff]
184	Formula	eta_b_X	4,6*[N_1x]*[Dim_X]/[L_I_zeff]	Building	4,6*[N_1x]*[Dim_X]/[L_I_zeff]
185	Formula	eta_b_Y	4,6*[N_1x]*[Dim_Y]/[L_I_zeff]	Building	4,6*[N_1x]*[Dim_Y]/[L_I_zeff]
186	Formula	R_h	1/[eta_h]-1/(2*[eta_h]^2)*(1-exp(-2*[eta_h]))	Building	1/[eta_h]-1/(2*[eta_h]^2)*(1-exp(-2*[eta_h]))
187	Formula	R_h	1	Building	1
188	Rule-set		R_h	Building	(1) if [eta_h]>0, then use Phenotype1_186 (2) else use Phenotype1_187
189	Formula	R_b_X	1/[eta_b_X]-1/(2*[eta_b_X]^2)*(1-exp(-2*[eta_b_X]))	Building	1/[eta_b_X]-1/(2*[eta_b_X]^2)*(1-exp(-2*[eta_b_X]))
190	Formula	R_b_Y	1/[eta_b_Y]-1/(2*[eta_b_Y]^2)*(1-exp(-2*[eta_b_Y]))	Building	1/[eta_b_Y]-1/(2*[eta_b_Y]^2)*(1-exp(-2*[eta_b_Y]))
191	Formula	R_b_X	1	Building	1
192	Formula	R_b_Y	1	Building	1

Id	Type	Identifier	Name	Product-model	Content
193	Formula	R_N	6,8*[N_1x]/(1+10,2*[N_1x])^(5/3)	Building	6,8*[N_1x]/(1+10,2*[N_1x])^(5/3)
194	Rule-set		R_b_X	Building	(1) if [eta_b_X]>0, then use Phenotype1_189 (2) else use Phenotype1_191
195	Rule-set		R_b_Y	Building	(1) if [eta_b_Y]>0, then use Phenotype1_190 (2) else use Phenotype1_192
196	Formula	R_x_quad_X	pi()^2/(2*0,1)*[R_N]*[R_h]*[R_b_X]	Building	pi()^2/(2*0,1)*[R_N]*[R_h]*[R_b_X]
197	Formula	R_x_quad_Y	pi()^2/(2*0,1)*[R_N]*[R_h]*[R_b_Y]	Building	pi()^2/(2*0,1)*[R_N]*[R_h]*[R_b_Y]
198	Formula	Q_0_quad_X	1/(1+0,9*(([Dim_X]+[Dim_Z])/[L_I_zeff])^0,63)	Building	1/(1+0,9*(([Dim_X]+[Dim_Z])/[L_I_zeff])^0,63)
199	Formula	Q_0_quad_Y	1/(1+0,9*(([Dim_Y]+[Dim_Z])/[L_I_zeff])^0,63)	Building	1/(1+0,9*(([Dim_Y]+[Dim_Z])/[L_I_zeff])^0,63)
200	Formula	S_X	0,46*(([Dim_X]+[Dim_Z])/[L_I_zeff])+10,58*(sqrt([Dim_X]*[Dim_Z])/[L_I_zeff])	Building	0,46*(([Dim_X]+[Dim_Z])/[L_I_zeff])+10,58*(sqrt([Dim_X]*[Dim_Z])/[L_I_zeff])
201	Formula	S_Y	0,46*(([Dim_Y]+[Dim_Z])/[L_I_zeff])+10,58*(sqrt([Dim_Y]*[Dim_Z])/[L_I_zeff])	Building	0,46*(([Dim_Y]+[Dim_Z])/[L_I_zeff])+10,58*(sqrt([Dim_Y]*[Dim_Z])/[L_I_zeff])
202	Formula	ny_0_X	[v_m_zeff]/[L_I_zeff]*1/(1,11*[S_X]^0,615)	Building	[v_m_zeff]/[L_I_zeff]*1/(1,11*[S_X]^0,615)
203	Formula	ny_0_Y	[v_m_zeff]/[L_I_zeff]*1/(1,11*[S_Y]^0,615)	Building	[v_m_zeff]/[L_I_zeff]*1/(1,11*[S_Y]^0,615)
204	Formula	ny_X	sqrt(([ny_0_X]^2*[Q_0_quad_X]+[Eigenfrequenz]^2*[R_x_quad_X])/([Q_0_quad_X]+[R_x_quad_X])	Building	sqrt(([ny_0_X]^2*[Q_0_quad_X]+[Eigenfrequenz]^2*[R_x_quad_X])/([Q_0_quad_X]+[R_x_quad_X])
205	Formula	ny_Y	sqrt(([ny_0_Y]^2*[Q_0_quad_Y]+[Eigenfrequenz]^2*[R_x_quad_Y])/([Q_0_quad_Y]+[R_x_quad_Y])	Building	sqrt(([ny_0_Y]^2*[Q_0_quad_Y]+[Eigenfrequenz]^2*[R_x_quad_Y])/([Q_0_quad_Y]+[R_x_quad_Y])

Id	Type	Identifier	Name	Product-model	Content
206	Formula	Spitzenwert_g	sqrt(2*log([Eigenfrequenz]*600))+0,6/sqrt(2*log([Eigenfrequenz]*600))	Building	sqrt(2*log([Eigenfrequenz]*600))+0,6/sqrt(2*log([Eigenfrequenz]*600))
207	Formula	Boenreaktionsfaktor_G_X	1+2*[Spitzenwert_g]*[I_v_zeff]*sqrt([Q_0_quad_X]+[R_x_quad_X])	Building	1+2*[Spitzenwert_g]*[I_v_zeff]*sqrt([Q_0_quad_X]+[R_x_quad_X])
208	Formula	Boenreaktionsfaktor_G_Y	1+2*[Spitzenwert_g]*[I_v_zeff]*sqrt([Q_0_quad_Y]+[R_x_quad_Y])	Building	1+2*[Spitzenwert_g]*[I_v_zeff]*sqrt([Q_0_quad_Y]+[R_x_quad_Y])
209	Formula	I_v_zeff	[f_I]*([Z_Eff]/10)^[e_I]	Building	[f_I]*([Z_Eff]/10)^[e_I]
210	Constant	c_pe	Kraftbeiwerte Druck=0,8 + Sog=0,5	Building	Value=1.3000
211	From Product-model (Derived)	Floor_Z_Level	Z-LEVEL	Floor	Z-LEVEL delivers the Z-LEVEL
212	Formula	z_u	[Floor_Z_Level]	Floor	[Floor_Z_Level]
213	Formula	Floor_h_G	[Floor_Above.Floor_Z_Level]-[Floor_Z_Level]	Floor	[Floor_Above.Floor_Z_Level]-[Floor_Z_Level]
214	Formula	z_m	[Floor_Z_Level]+[Floor_h_G]/2	Floor	[Floor_Z_Level]+[Floor_h_G]/2
215	Formula	z_o	[Floor_Z_Level]+[Floor_h_G]	Floor	[Floor_Z_Level]+[Floor_h_G]
216	Formula	q_u	([f_v_m]*[v_ref_0]*([z_u]*10)^[e_v_m])^2/1600	Floor	([f_v_m]*[v_ref_0]*([z_u]*10)^[e_v_m])^2/1600
217	Formula	q_m	([f_v_m]*[v_ref_0]*([z_m]*10)^[e_v_m])^2/1600	Floor	([f_v_m]*[v_ref_0]*([z_m]*10)^[e_v_m])^2/1600
218	Formula	q_o	([f_v_m]*[v_ref_0]*([z_o]*10)^[e_v_m])^2/1600	Floor	([f_v_m]*[v_ref_0]*([z_o]*10)^[e_v_m])^2/1600
219	Formula	Geschoss-last_Wind_F_X	[Floor_h_G]/6*([q_u]+4*[q_m]+[q_o])+[Dim_Y]*[Boenreaktionsfaktor_G_X]*[c_pe]	Floor	[Floor_h_G]/6*([q_u]+4*[q_m]+[q_o])+[Dim_Y]*[Boenreaktionsfaktor_G_X]*[c_pe]
220	Formula	Geschoss-last_Wind_F_Y	[Floor_h_G]/6*([q_u]+4*[q_m]+[q_o])+[Dim_X]*[Boenreaktionsfaktor_G_Y]*[c_pe]	Floor	[Floor_h_G]/6*([q_u]+4*[q_m]+[q_o])+[Dim_X]*[Boenreaktionsfaktor_G_Y]*[c_pe]

Id	Type	Identifier	Name	Product-model	Content
221	Formula	Geschoss-last_Wind_M_X	[Geschoss-last_Wind_F_X]*[Dim_Y]/10	Floor	[Geschoss-last_Wind_F_X]*[Dim_Y]/10
222	Formula	Geschoss-last_Wind_M_Y	[Geschoss-last_Wind_F_Y]*[Dim_X]/10	Floor	[Geschoss-last_Wind_F_Y]*[Dim_X]/10
223	Formula	Geschoss-last_Wind_F_X	0,5*[Floor_h_G]/6*([q_u]+4*[q_m]+[q_o])*[Dim_Y]*[Boenreaktionsfaktor_G_X]*[c_pe]	Floor	0,5*[Floor_h_G]/6*([q_u]+4*[q_m]+[q_o])*[Dim_Y]*[Boenreaktionsfaktor_G_X]*[c_pe]
224	Formula	Geschoss-last_Wind_F_Y	0,5*[Floor_h_G]/6*([q_u]+4*[q_m]+[q_o])+[Dim_X]*[Boenreaktionsfaktor_G_Y]*[c_pe]	Floor	0,5*[Floor_h_G]/6*([q_u]+4*[q_m]+[q_o])+[Dim_X]*[Boenreaktionsfaktor_G_Y]*[c_pe]
225	Rule-set		Geschosslast	Floor	(1) if [Floor_Above]>0, then use Phenotype1_219, Pheno-type1_220 (2) else use Phenotype1_223, Phenotype1_224
226	Komplex table	Psi0;Psi1;Psi2;	Kombinationsbeiwerte	Building	complex content
227	Constant	Kategorie_Nutzung	B	Building	Value=B
228	Constant	gamma_g	Jan 35	Building	Value=1.3500
229	Constant	gamma_q	Jan 50	Building	Value=1.5000
230	Komplex table	Psi0_Wind;Psi1_Wind;Psi2_Wind;	Kombinationsbeiwer-te_Wind	Building	complex content
231	Rule-set		Floor_H_G	Floor	(1) if [Floor_Above]>0 , then use Phenotype1_213 (2) else use Phenotype1_232
232	Constant	Floor_h_G	0	Floor	Value=0.0000
233	Formula	Flaeche	[dx]*[dy]	Element	[dx]*[dy]
234	Sensiti-vity selection		Stress	Slab	Sensitivity value: ([Stress_M11_Max]+[Stress_M22_Max])/220 Specifity value: 1
235	From Product-model (Derived)	Stress_M11_Max	STRESS_RESULTANTS	Slab	STRESS_RESULTANTS Delivers parameter dependent Stress resultants Resultent_Type Max_Min
236	From Product-model (Derived)	Stress_M22_Max	STRESS_RESULTANTS	Slab	STRESS_RESULTANTS Delivers parameter dependent Stress resultants Resultent_Type Max_Min

Id	Type	Identifier	Name	Product-model	Content
237	From Product-model (Derived)	SumVolumen_Beton	Volumen_Beton	Building	SUM_ELEMENTS Sum function, summarizes values in all elements Value Type
239	Formula	Fitness	1/[Bewertung]	Building	1/[Bewertung]
240	Komplex table	omega_1;xi;zeta;epsilon_c2;epsilon_s1;sigma_s1_d;omega_2;sigma_s2_d;	omega Tabelle 4.2, C12/15-C50/60	Element	complex content
241	Formula	A_c	[b]*[h]*10000	Slab	[b]*[h]*10000
242	From Product-model (Basic)	h_Decke	EXT_D	Slab	EXT_D delivers the total depth of the Slab
243	From Product-model (Basic)	b_Decke_x	EXT_X	Slab	EXT_X delivers the total x-extent of the Slab EXT_X delivers the total x-extent of the element EXT_X delivers the total x-extent of the element
244	From Product-model (Basic)	b_Decke_y	EXT_Y	Slab	EXT_Y delivers the total y-extent of the Slab EXT_Y delivers the total y-extent of the element EXT_Y delivers the total y-extent of the element
245	Constant	b	b_Decke=1.0m	Slab	Value=1.0000 m
246	Formula	h	[h_Decke]	Slab	[h_Decke]
247	Formula	W	[b]*[h]^2/6	Slab	[b]*[h]^2/6
248	Formula	M_sd	max(max(abs([Stress_M11_Min]);abs([Stress_M22_Min]));max(abs([Stress_M11_Max]);abs([Stress_M22_Max])))	Slab	max(max(abs([Stress_M11_Min]);abs([Stress_M22_Min]));max(abs([Stress_M11_Max]);abs([Stress_M22_Max])))
249	Formula	mue_Eds_rechnerisch	([M_sds]/1000)/([b]*([d]/100)^2*[f_cd])	Slab	([M_Eds]/1000)/([b]*([d]/100)^2*[f_cd])

Id	Type	Identifier	Name	Product-model	Content
250	Proof selection		Bending Design - Compressive Reinforcement	Slab	Proof condition: [erf_a_s1_M]<=[max_a_sl_M]
251	Formula	d	[h]*100-[c_nom]/10-[d_sl_1]/2/10	Slab	[h]*100-[c_nom]/10-[d_sl_1]/2/10
252	Formula	f_cd	[alpha_C]*[f_ck]/[gamma_c_Ortbeton]	Building	[alpha_C]*[f_ck]/[gamma_c_Ortbeton]
253	Constant	alpha_C	Beiwert für Normalbeton	Building	Value=0.8500
254	Komplex table	gamma'_c;gamma_c_Ortbeton;gamma_c_Fertigteil;	Teilsicherheitsbeiwerte fuer Beton	Building	complex content
255	Komplex table	f_ck;f_ck_cube;f_cm;f_ctm;f_ctk_005;f_ctk_095;E_cm;	Festigkeitskennwerte von Normalbeton	Building	complex content
256	Constant	Betonfestigkeitsklasse	C30/37	Building	Value=Undefined
257	Formula	erf_A_sl	1/[sigma_s1_d]*([omega_1]*[b]*([d]/100)*[f_cd]+[N_Ed]/1000)*10^4	Slab	1/[sigma_s1_d]*([omega_1]*[b]*([d]/100)*[f_cd]+[N_Ed]/1000)*10^4
258	Constant	N_Ed	0	Slab	Value=0.0000
259	Constant	Kostenindex_Stahl	€/t	Building	Value=1000.0000 €/t
260	Constant	Kostenindex_Beton	€/m³	Building	Value=140.0000 €/m³
262	Formula	Volumen_Stahl	[erf_A_sl]*[dx]*[dy]*2/10000	Slab	[erf_A_sl]*[dx]*[dy]*2/10000
263	Formula	Grundflaeche	0	Element	0
264	Formula	Grundflaeche	[dx]*[dy]	Slab	[dx]*[dy]
265	Formula	Konstruktionsflaeche	0	Element	0
266	Formula	Konstruktionsflaeche	[dx]*[dy]	Vertical element	[dx]*[dy]
267	Formula	Volumen_Stahl	0	Element	0
268	From Product-model (Derived)	SumVolumen_Stahl	Volumen_Stahl	Building	SUM_ELEMENTS Sum function, summarizes values in all elements Value Type
269	Constant	Dichte_Stahl	New	Building	Value=7.0000 t/m³

Id	Type	Identifier	Name	Product-model	Content
270	From Product-model (Derived)	SumGrund-flaeche	SUM_ELEMENTS	Building	SUM_ELEMENTS Sum function, summarizes values in all elements Value Type
271	From Product-model (Derived)	SumKon-struktions-flaeche	SUM_ELEMENTS	Building	SUM_ELEMENTS Sum function, summarizes values in all elements Value Type
272	Formula	Nutzflaeche	[SumGrundflaeche]-[SumKonstruktions-flaeche]	Building	[SumGrundflaeche]-[SumKonstruktionsflaeche]
273	Formula	Material-kosten	[Kostenin-dex_Beton]*[SumVolumen_Beton]+[Kostenindex_Stahl]*[SumVolumen_Stahl]*[Dichte_Stahl]	Building	[Kostenin-dex_Beton]*[SumVolumen_Beton]+[Kostenindex_Stahl]*[SumVolumen_Stahl]*[Dichte_Stahl]
274	Constant	Ertragsin-dex	Ertrag /m² über 12 Nutzungsjahre	Building	Value=3600.0000 €/m²
275	Formula	Gesamter-trag	[Ertragsin-dex]*[Nutzflaeche]	Building	[Ertragsindex]*[Nutzflaeche]
276	Constant	Baukosten-index	Gesamtkos-ten/Materialkosten	Building	Value=4.0000
277	Formula	Gesamtkos-ten	[Baukostenin-dex]*[Materialkosten]	Building	[Baukostenin-dex]*[Materialkosten]
278	Formula	Bewertung	[Gesamter-trag]/[Gesamtkosten]/([SumStraffakto-ren]+1)	Building	[Gesamter-trag]/[Gesamtkosten]/([SumStraffaktoren]+1)
279	Constant	d_sl_1	Stabdurchmesser = 10mm	Slab	Value=10.0000 mm
280	Formula	c_nom	[c_min]+[delta_c]	Element	[c_min]+[delta_c]
281	Formula	c_min	[d_sl_1]	Element	[d_sl_1]
282	Komplex table	c_min_tmp;c_min_Spannglied_tmp;delta_c;	(Kopie) c_min und delta_c	Element	complex content
283	Constant	Expositi-onsklasse	XC1	Building	Value=XC1
284	Formula	Q_DECKE	[q_k]	Slab	[q_k]
285	From Product-model (Derived)	Stress_M11_Min	STRESS_RESULTANTS	Slab	STRESS_RESULTANTS Delivers parameter dependent Stress resultants Resultent_Type Max_Min

Id	Type	Identifier	Name	Product-model	Content
286	From Product-model (Derived)	Stress_M22_Min	STRESS_RESULTANTS	Slab	STRESS_RESULTANTS Delivers parameter dependent Stress resultants Resultent_Type Max_Min
287	Constant	mue_Eds_lim	0.2960	Slab	Value=0.2960
288	Formula	Straffaktor	(max(1;[mue_Eds_rechnerisch]/[mue_Eds_lim])-1)*10	Slab	(max(1;[mue_Eds_rechnerisch]/[mue_Eds_lim])-1)*10
289	Formula	mue_Eds	min([mue_Eds_rechnerisch];[mue_Eds_lim])	Slab	min([mue_Eds_rechnerisch];[mue_Eds_lim])
290	From Product-model (Derived)	SumStraffaktoren	SUM_ELEMENTS	Building	SUM_ELEMENTS Sum function, summarizes values in all elements Value Type
291	From Product-model (Derived)	Stress_N1_Min	STRESS_RESULTANTS	Column	STRESS_RESULTANTS Delivers parameter dependent Stress resultants Resultent_Type Max_Min
292	Formula	Straffaktor	max(1;([N_Sd_Stuetze]/[N_Rd_Stuetze]))-1	Column	max(1;([N_Sd_Stuetze]/[N_Rd_Stuetze]))-1
293	Formula	N_Rd_Stuetze	[f_ck]	Column	[f_ck]
294	Formula	N_Sd_Stuetze	abs([Stress_N1_Min])/1000/[dx]/[dy]	Column	abs([Stress_N1_Min])/1000/[dx]/[dy]
295	From Product-model (Derived)	Stress_M3_Min	STRESS_RESULTANTS	Column	STRESS_RESULTANTS Delivers parameter dependent Stress resultants Resultent_Type Max_Min
296	From Product-model (Derived)	Stress_M2_Min	STRESS_RESULTANTS	Column	STRESS_RESULTANTS Delivers parameter dependent Stress resultants Resultent_Type Max_Min
297	From Product-model (Derived)	Stress_M2_Max	STRESS_RESULTANTS	Column	STRESS_RESULTANTS Delivers parameter dependent Stress resultants Resultent_Type Max_Min

Id	Type	Identifier	Name	Product-model	Content
298	From Product-model (Derived)	Stress_M3_Max	STRESS_RESULTANTS	Column	STRESS_RESULTANTS Delivers parameter dependent Stress resultants Resultent_Type Max_Min
299	Formula	M_Stuetze	max(max(abs([Stress_M2_Min]);abs([Stress_M3_Min]));max(abs([Stress_M2_Max]);abs([Stress_M3_Max])))	Column	max(max(abs([Stress_M2_Min]);abs([Stress_M3_Min]));max(abs([Stress_M2_Max]);abs([Stress_M3_Max])))
300	Constant	N_sd	0	Slab	Value=0.0000 kN
301	Formula	M_sds	[M_sd]-[N_sd]*[z_s]	Slab	[M_sd]-[N_sd]*[z_s]
302	Formula	z_s	0	Slab	0

9.3 Anhang C: Bildungsvorschrift

S:=Bauwerk
Bauwerk:=GEB[" Raster Kerne Stockwerkgruppen "]
Raster:=FELDER_X=" Felder ";FELDER_Y=" Felder ";RASTER_X=" Rastergroesse ";RASTER_Y="Rastergroesse" Rastergroesse:=[5;9;0.05] Felder:=[4;6;1]
Kerne:=|Kern|Kern Kern
Kern:=KERN[KFELD X PRZ=" KernfeldX ";KFELD Y PRZ=" KernfeldY ";KFELDER_X_PRZ=" KernfelderX ";KFELDER_Y_PRZ=" KernfelderY ";KMAX_STW=" [2;20;1] ";]
KernfeldX:=[0;1;0.01] KernfeldY:=[0;1;0.01]
KernfelderX:=[0;0.4;0.01] KernfelderY:=[0;0.4;0.01] Stockwerkgruppen:=Stockwerkgruppe_Ende|Stockwerkgruppe Stockwerkgruppe_Ende|Stockwerkgruppe Stockwerkgruppe_Outrigger Stockwerkgruppe Stockwerkgruppe Ende|Stockwerkgruppe Stockwerkgruppe Outrigger Stockwerkgruppe Stockwerkgruppe_Outrigger Stockwerkgruppe Stockwerkgruppe_Ende
Stockwerkgruppe:=SWG[ANZ_STW=" [1;6;1] ";H_STW=" [3;5;.1] ";OUTRIGGER=2.0;" Deckensystem Ssg1 Ssg2 Ssg3 Ssg4 "]□Stockwerkgruppe_Ende:=SWG[ANZ_STW=0;H_STW=" [3;5;.1] ";OUTRIGGER=2.0;" Deckensystem Ssg1 Ssg2 Ssg3 Ssg4 "]□Stockwerkgruppe_Outrigger:=SWG[ANZ_STW=1;H_STW="[2;4;.2] ";OUTRIGGER=1.0;" Deckensystem Ssg1 Ssg2 Ssg3 Ssg4 "]□Deckensystem:=Fd|Uzd
Fd:=DS=FD;H_D=" [0.2;0.4;0.01] "
Uzd:=DS=UZD;H_D=" [0.2;0.4;0.01] ";H_UZ=" [0.2;0.4;0.01] ";B_UZ=" [0.2;0.3;0.01] ";DIR_UZ=" Dir_Uzd " Dir_Uzd:=x"|"y" Ssg1:=SSG[NAME=x-edge;DX=" SsgDim ";DY=" SsgDim ";]
Ssg2:=SSG[NAME=y-edge;DX=" SsgDim ";DY=" SsgDim ";]
Ssg3:=SSG[NAME=corner;DX=" SsgDim ";DY=" SsgDim ";]
Ssg4:=SSG[NAME=inside;DX=" SsgDim ";DY=" SsgDim ";]
SsgDim:=[0.20;0.60;0.01]

Literaturverzeichnis

[ADA03] Adamy, J.: *Fuzzy-Logik, Neuronale Netze und Evolutionäre Algorithmen*, Vorlesungsumdruck, Technische Universität Darmstadt, 2003

[ALB02] Albert, A.: *Wissensbasiertes Entwerfen und Bemessen von Tragwerken unter Einsatz von Fuzzy-Methoden*; Dissertation, TU-Darmstadt, Fortschritt-Berichte VDI, VDI-Verlag, Düsseldorf, 2002

[ARC01] Arciszewski, T., De Jong, K.: *Evolutionary Computation in Civil Engineering: Research Frontiers*, invited article, B.H.V. Topping, (Editor), Civil and Structural Engineering Computing, 2001

[BÄC93] Bäck, T.: *Optimal Mutation Rates in Genetic Search*, Proceedings of the 5th international conference on Genetic Algorithms, San Mateo, USA, Morgan Kauffmann, 1993

[BÄC97] Bäck, T., Fogel, D. B., Michalewicz, Z.: *Handbook of Evolutionary Computation*, Institute of Physics Publishing, Philadelphia, and Oxford University Press, Oxford, 1995

[BAL01] Balzert, H.: *Lehrbuch der Software-Technik*, Spektrum, Heidelberg, 2001

[BEI06] Beierle, C., Kern-Isberner, G.: *Methoden wissensbasierter Systeme*, 3. Auflage, Vieweg, Wiesbaden, 2006

[BET02] *Beton Kalender*, Ernst & Sohn, 2002

[BIB93] Bibel, W.: *Wissensrepräsentation und Inferenz – eine grundlegende Einführung*, Vieweg, Wiesbaden, 1993

[BLE02] Bletzinger, K.-U., Wüchner, R., Daoud, F.: *Merging Form Finding and Shape Optimization Methods*, in Proceedings of the Fifth World Congress on Computational Mechanics (WCCM V), Vienna University of Technology, Austria, 2002

[BOI09] BOINC - Berkeley Open Infrastructure for Network Computing, *http://boinc.ssl.berkeley.edu/*, University of California, USA (Stand: Januar 2010)

[BOR07] Borrmann, A. et al: *An Octree-based Implementation of Directional Operators in a 3D Spatial Query Language for Building Information Models*, in Proceedings to the 24th W78 conference 2007, Maribor, Slovenja, 2007

[BUC83] Buchanan, B. G. et al.: *Constructing an Expert System.* In: Hayes-Roth, F., Waterman, D. A. und Lenat, D. B. (Hrsg.): Building Expert Systems. Reading, 1983

[CLA05] Clarkson, J., Eckert, C.: *Design Process Improvement, A review of current practice*, Springer, Berlin, 2005

[DAR1894] Darwin, C.: *Die Entstehung der Arten*, Kröner-Verlag, Leipzig, 1894

[DEB95] Deb, K., Goyal, M.: *Optimizing engineering designs using a combined genetic search*, Proceedings of the Sixth International Conference on Genetic Algorithms, Morgan Kaufmann, San Francisco, USA, 1995

[DIN1055-4] DIN 1055-4 (2005-03): *Windlasten*, Beuth, Berlin, 2005

[DIN4149] DIN 4149 (2005-04): *Bauten in deutschen Erdbebengebieten: Lastannahmen, Bemessung und Ausführung üblicher Hochbauten,* Beuth, Berlin, 2005

[FOG00] Fogel, D. B.: *Evolutionary Computation – Toward a new Philosophy of Machine Intelligence*, IEEE Press, Piscantaway, USA, 2000

[FOG66] Fogel, L. J.: *Artificial Intelligence through Simulated Evolution*, John Wiley, New-York, 1966

[GAR92] Garrett, J. H., Hakim, M.M.: *An Object-Oriented Model of Engineering Design Standards*, Journal of computing in civil engineering, 6(3), 323-347, 1992

[GEY07] Geyer, P.: *Embedding Optimization in the Design Process of Buildings – A Hall Example*, in Proceedings to the 24th W78 conference 2007, Maribor, Slovenja, 2007

[GÖR95] Görz, G.: *Einführung in die künstliche Intelligenz*, Addison-Wesley, Bonn, 1995

[GRI02] Grierson, D. E., Khajehpour, S.: *Method for Conceptual Design Applied to Office Buildings*, ASCE J. of Computing in Civil Engineering, Nr. 16, 2002

[GRI07] Grierson, D. E.: *Multicriteria Decision Making in n-d*, in Proceedings to the 24th W78 conference 2007, Maribor, Slovenja, 2007

[HAL99] Hansheng, L., Lishan, K.: *Balance between Exploration and Exploitation in Genetic Search*, Wuhan University Journal of Natural Sciences, Vol. 4 No.1, Wuhan, China, 1999

[HAR96] Hartshorn, C. et al. (edts.): *Collected Papers of C. Sanders Pierce, Band 2*, Harvard University Press, Cambridge, 1996

[HEM02] Hemaspaandra, L., Torenvliet, L.: *Theory of Semi-Feasible Algorithms*, Springer, Berlin, 2002

[HIN96] Hinterding, R., Michalewicz, Z.: *Self-Adapted Genetic Algorithm for Numeric Functions*, in *Parallel Problem Solving from Nature*, Springer, Berlin, 1996

[HOL07] Holschemacher, K., Klug, Y.: *Lastannahmen nach neuen Normen*, Bauwerk, Berlin, 2007

[HOL75] Holland, J. H.: *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975

[HOL96] Holéwik, P.: *Ein objektorientiertes Wissensrepräsentationsmodell zur Entwicklung computerunterstützter Nachweissysteme im Bauwesen*, Dissertation, Technisch-wissenschaftliche Mitteilung Nr. 96-7, Institut für konstruktiven Ingenieurbau, Bochum, 1996

[JOS06] Joseph, L. M., Poon, D., Shaw-Song, S.: *Taipei 101*, Structure Magazine 06/2006, C³Ink, Reedsburg, 2006

[KAR05] Karczewski, B.: *Statische und dynamische Berechnung von Aussteifungssystemen im Stahlbetonhochbau*, Diplomarbeit, Universität Duisburg-Essen, 2006

[KNU64] Knuth, D. E.: *Backus Normal Form versus Backus Naur Form*, Communications of the ACM 7, ACM Press, New York, 1964

[KOE03] König, G., Liphardt, S.: *Hochhäuser aus Stahlbeton*, in Beton-Kalender 2003, Ernst & Sohn, Berlin 2003

[KOZ92] Koza, J. R.: *Genetic Programming – On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, 1992

[KOZ94] Koza, J. R.: *Genetic Programming II – Automatic Discovery of Reusable Programs*, MIT Press, Cambridge, 1994

[LAN02] Langdon, W. B., Poli, R.: *Foundations of Genetic Programming*, Springer, Berlin, 2002

[LAU04] Laubach, A.: *Anpassungsfähige Hochhaustragwerke : Machbarkeit und Optimierung*, Dissertation, Univ. Leipzig, Norderstedt, 2004

[LEI98] Leinecker, R. C., Archer, T.: *Die Visual C++ 6 Bibel*, Mitp, Bonn, 1998

[LIP06] Liphardt, S.: *Entwicklung von Hochhaustragwerken*, Tiefbau-Magazin 01/2006, Erich Schmidt Verlag, München, 2006

[MAJ06] Majeed, H., Ryan, C.: *A Less Destructive, Context-Aware Crossover Operator for GP*, in Genetic Programming, Procedings to the 9th European Converence EuroGP, Springer, Berlin, 2006

[MON94] Montana, D. J.: *Strongly Typed Genetic Programming*, BBN Technical Report #7866, Cambridge, 1994

[PAH00] Pahl, P. J., Damrath, R.: *Mathematische Grundlagen der Ingenieurinformatik*, Springer, Berlin, 2000

[PHO01] Phocas, M. C.: Tragwerke für den Hochhausbau. Verlag Ernst & Sohn, 2001

[POH99] Pohlheim, H.*: Evolutionäre Algorithmen*, Springer, Berlin, 1999

[POK07] Pokojski, J.: *Integration of Knowledge Based Approach and Multicriteria Optimization in Engineering Design*, in Proceedings to the 24th W78 conference 2007, Maribor, Slovenja, 2007

[POL01] Poli, R., McPhee, N. F.: *Exact GP Schema Theory for Headless Chicken Crossover and Subtree Mutation*, IEEE Press, Seoul, Korea, 2001

[PUL03] Pullmann, T., Arciszewski, T., Schnellenbach-Held, M. et al.: *Structural Design of Reinforced Concrete Tall Buildings: Evolutionary Computation Approach using Fuzzy Sets*, in Intelligent Computing in Engineering, Proceedings of the 10th G-ICE Workshop, Delft, Netherlands, 2003

[RAF99] Rafiq, M. Y., Bugmann, G., Easterbrook, D. J.: *Building Concept Generation Using Genetic Algorithms Integrated with Neural Networks*, in Artificial Intelligence in Structural Engineering, Wydawnictwa Naukowo-Techniczne, Warschau, 1999

[RAM00] Ramm, E., Schwarz, S., Kemmler, R.: *Advances in Structural Optimization Including Nonlinear Mechanics*, Proc. of 'ECCOMAS 2000', Barcelona, Spain, 2000

[REC73] Rechenberg, I.: *Evolutionsstrategie – Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Frommann-Verlag, Stuttgart, 1973

[ROB07] Robertson, L. E., See, S.: *Shanghai World Financial Center*, Structure Magazine 06/2007, C³Ink, Reedsburg, 2007

[ROS96] Rosca, J. P., Ballard, D. H.: *Discovery of Subroutines in Genetic Programming*, MIT Press, Cambridge, 1996

[ROZ97] Rozenberg, G., Salomaa, A.: *Handbook of Formal Languages. Volume 1. Word, Language, Grammar,* Springer, Berlin, 1997

[SAR00] Sarma, K. C., Adeli, H.: *Fuzzy Discrete Multicriteria Cost Optimization of Steel Structures*, Journal of Structural Engineering, Vol. 126, Nr. 11, 2000

[SCHN91] Schnellenbach, M.: *Wissensbasierte Integration und Steuerung computergestützter Entwurfsprozesse im Stahlbetonbau*, Dissertation, Ruhr-Universität Bochum, Institut für konstruktiven Ingenieurbau, 1991

[SCHW95] Schwefel, H.-P.: *Evolution and Optimum Seeking*, John Wiley, New-York, 1995

[SHA03] Shaw, D., Miles, J., Gray, A.: *Genetic Programming within Civil Engineering: A Review*, in Intelligent Computing in Engineering, Proceedings of the 10th G-ICE Workshop, Delft, Netherlands, 2003

[SMI85] Smith, B. C.: *Proloque to „Reflection and Semantics in a Procedural Language“*, in Readings in Knowledge Representation, Morgan Kaufmann, Los Altos, 1985

[SOL66] Solmonoff, R. J.: *Some Recent Work in Artificial Intelligence*, in Procedures of the IEEE, Vol.54, 1966

[TAC94] Tackett, W. A.: *Recombination, Selection, and the Genetic Construction of Computer Programs*, PhD thesis, University of Southern Carolina, USA, 1994

[THI04] Thierauf, G.: *Benutzerhandbuch B&B,* Universität Duisburg-Essen, 2004

[THI96] Thierauf, G.: *Direct Search, Stochastic Search and Darwinian Methods in Structural Optimization and Interactions with Parallel Computing*, Advances in Computational Structures Technology, Civil-Comp. Press, Edinburgh, 1996

[WAT68] Watson, J. D.: *Double Helix*, Touchstone, New-York, 1968

[WIK07/1] *Wikipedia – Schlagwort "Wissensrepräsentation"*, de.wikipedia.org, Stand 09/2007

[YAN02] Yang, Y., Soh, C. K.: *Automated Optimum Design of Structures using Genetic Programming*, Computers and Structures, Volume 80, Issues 18-19, Pergamon, 2002

[ZEL08] Zeller, T.: *OpenVPN Kompakt*, Bomots, Stiring-Wendel, 2008

Bildnachweis

Abbildung 2-12 Quelle: Flickr
http://www.flickr.com/photos/loop_oh/5277632633/in/set-72157603794444821/

Urheber: Rupert Ganzer

Lizenz: Creative Commons (BY-ND)
http://creativecommons.org/licenses/by-nd/2.0/deed.de

Abbildung 2-13 Quelle: Flickr
http://www.flickr.com/photos/vike/5102172282/in/set-72157625210320418/

Urheber: Rubén Vique

Lizenz: Creative Commons (BY),
http://creativecommons.org/licenses/by/2.0/deed.de

Abbildung 2-21 Quelle: Flickr
http://www.flickr.com/photos/yakobusan/472204557/in/photostream/

Urheber: Jakob Montrasio

Lizenz: Creative Commons (BY),
http://creativecommons.org/licenses/by/2.0/deed.de

Abbildung 2-22 Quelle: Flickr
http://www.flickr.com/photos/yakobusan/472203355/in/photostream/

Urheber: Jakob Montrasio

Lizenz: Creative Commons (BY),
http://creativecommons.org/licenses/by/2.0/deed.de

Abbildung 2-23 Quelle: Wikimedia Commons
http://commons.wikimedia.org/wiki/File:Taipei_101_2009_amk.jpg

Urheber: Matthias Trischler, Reichenbach an der Fils

Mit freundlicher Genehmigung