استفاده از الگوریتم ژنتیک در مسئله‌ی بهینه‌سازی درخت پوشای ارتباطی با چند تابع هدف

سید علی نواب کاشانی
کارشناس مهندسی نرم‌افزار کامپیوتر
sanavab@gmail.com

سید مهدی وحیدی پور
عضو هیأت علمی دانشگاه، گروه کامپیوتر، دانشکده مهندسی
دانشگاه کاشان، کاشان، ایران
vahidipour@kashanu.ac.ir

چکیده: مسائل بهینه سازی در ارتباط با شبکه‌های ارتباطی توسط بسیاری از محققان شبکه مورد بررسی قرار گرفته است. بهینگی اتصال شبکه از لحاظ پارامترهای استاندارد کیفیتِ خدمت (QOS) از مهمترین مسائل می‌باشد. اخیراً الگوریتم‌های ژنتیک در زمینه‌های تحقیقاتی مذکور کاربرد زیادی داشته‌اند. همچنین الگوریتم‌های ژنتیک با چند تابع هدف از لحاظ توانایی بهینه‌سازی در مسائل جهان واقعی مورد توجه زیادی قرار گرفته‌اند. در این مقاله ابتدا مسئله‌ی درخت پوشای ارتباطی بهینه (OCST) به صورت چند تابع هدفی تعریف شده است. کمینه کردن هزینه مکانی و متوسط تاخیر پیام، اهداف مورد نظر است. سپس این مسئله با استفاده از الگوریتم ژنتیک مبتنی بر درخت های پوشا مورد حل و بررسی قرار گرفته است. درانتها آزمایشات مختلفی با دو روش از الگوریتم‌های تکاملی چندتابعی، SPEA-II و NSGA-II و با استفاده از واسط PISA انجام و مقایسه شده است.

واژه های کلیدی : الگوریتم ژنتیک، بهینه‌سازی با چند تابع هدف، درخت پوشای ارتباطی بهینه (OCST)، SPEA-II ، NSGA-II ، PISA.

۱- مقدمه

شبکه‌های کامپیوتری محلی عموماً به عنوان زیر ساختار ارتباطی استفاده می‌شوند که نیازهای ارتباطی کاربران را در محیط‌های محلی برآورده می‌کنند. برای این منظور با درنظر گرفتن شبکه‌ی ارتباطی به صورت یک گرافِ کامل، به مسیرهای متصل و بدون دور بین گره‌ها نیاز است. بنابراین فقط درخت‌های پوشا می‌توانند به عنوان معرفی مسیر ارتباطی در نظر گرفته شوند[۱]. همچنین به عنوان طراحی ارتباطات این سیستمهای شبکه کامپیوتری، یکی از معیارهای مهم بهینه کردن کارایی می باشد. مؤلفه های کارایی مانند هزینه، متوسط تاخیر پیام، ترافیک، اعتماد پذیری و غیره می‌باشند[۲،۱۴]. کارایی این سیستم ها بسیار پر اهمیت می باشد و به صورت عمده منوط به مسیر ارتباطی می‌باشد[۲]. الگوریتم های ژنتیک با توجه به پتانسیل تکنیک‌های بهینه‌سازی‌اش، توجه زیادی را در بهینه‌سازی شبکه‌ای به سمت خود جذب کرده و اغلب برای مسائل جهان واقعی نیز مورد استفاده قرار می-گیرد. همچنین تحقیقات بسیاری بر روی الگوریتم ژنتیک با مسائل چند تابع هدفی موجود می‌باشد[۳،۴،۵].

در این مقاله سعی بر آن شده است که مسئله درخت پوشای ارتباطی بهینه (OCST) در شبکه‌های محلی به صورت چند تابع هدفی تعریف و حل شود. تلاش برای کمینه کردن دو پارامتر هزینه مکانی و متوسط تاخیر پیام در توابع هدف دیده شده‌است. از آن‌رو که فقط درخت‌های پوشا می‌توانند برای ارتباطِ موثر استفاده شوند[۶, ۱۷]، دراین مقاله الگوریتم ژنتیک مبتنی بر درخت های پوشا را طراحی و به خدمت گرفته شد.

بحث کلیدی در طراحی الگوریتم ژنتیک، نحوه‌ی نمایش فضای جواب مسئله است[۷]. در الگوریتم ژنتیک ساده گلدبرگ[۸]، عموماً از فضای نمایش ثابت شامل رشته های دودویی استفاده می‌شود که کروموزوم خطی نامیده می‌شود. این نمایش فقط برای مسائلی که به صورت طبیعی و مستقیم روی رشته ها و بردار های دودویی تعریف می شود، جواب می دهد[۷]. به هر حال برای مسائلی، مانند OCST، که جواب های ممکن آنها، شکل‌های پیچیده، مانند درخت، دارند استفاده از کروموزوم‌های خطی، غیر طبیعی و غیر مؤثر واقع می‌شوند. بنابراین در ادامه‌ی تحقیقات، چندین کدینگ درختی جهت نمایش درختهای پوشا در الگوریتم‌های ژنتیک پیشنهاد شده‌است، مانند: عدد Prüfer[۹]، کدینگ Predecessor [۱۰]، نمایش درختی مبتنی بر فاکتورگیری معین [۱۱]، نمایش گره و یال پالمر[۱۲, ۱۳]. در این مقاله ایده عدد Prüfer به دلیل کارایی بالا و در عین حال سادگی در پیاده‌سازی برای نمایش درخت پوشا[۲, ۶]، مورد استفاده قرار می‌گیرد. از دیگر خصوصیات این کدینگ، قابلیت تناظر یک‌به‌یک جهت نشان دادن همه درختهای پوشای ممکن می باشد، که با استفاده از آن، حافظه برداری مورد نیاز برای مسئله با n گره ، n-2 می‌باشد[۶ , ۱۵].

هدف این مقاله رسیدن به جواب‌هایی جهت کمینه‌کردن همزمان هزینه انتقال مکانی و متوسط تاخیر پیام از پارامترهای استاندارد کیفیتِ خدمت، در شبکه‌های ارتباطی می‌باشد.

در قسمت‌های بعدی، ابتدا در خصوص سابقه و معرفی مسئله‌ی OCST توضیحاتی آورده می‌شود و تابع‌های هدف را تعریف می‌کنیم و بعد از به اصول بهینه‌سازی توسط الگوریتم‌های تکاملی با چند تابع هدف می‌پردازیم. در بخش بعدی با سفارشی کردن بهینه‌گر روی مسئله‌ی خاص خودمان، الگوریتم ژنتیک مبتنی بر درخت‌های پوشا را مطرح می‌کنیم و با ذکر مثال به تشریح بیشتر موضوع پرداخته می‌شود. بعد از آن واسطی که بوسیله‌ی آن روش‌های مختلف بهینه‌سازی روی مسئله‌ی مذکور امتحان و مقایسه شده‌است را معرفی می‌کنیم و در انتها، یکسری آزمایشات و نتایج گواه بر رسیدن به جواب‌های بهینه آورده می‌شود.

۲- توضیح مسئله و تابع‌های هدف

مسئله‌ی پیدا کردن درخت پوشایی که هزینه انتقال مجموعه داده شده از در خواست‌های ارتباطی بین n محل را کمینه می کند، اولین بار توسط Hu در سال ۱۹۷۴ [۱۶]، مطرح و به عنوان مسئله OCST شناخته می‌شود. در اینجا مسئله OCST به یک مسئله چند تابع هدفی ارتقا پیدا کرده‌است. به این صورت که درخت پوشا دو پارامتر هزینه انتقال مکانی و متوسط تأخیر پیام، را به طور همزمان کمینه می‌کند. توضیح اینکه این پارامترها ازمؤلفه‌های کارایی در استاندارد کیفیت خدمت (QOS) می‌باشد.

این مسئله که مستقیماً روی طراحی شبکه های ارتباطی درختی بوجود آمده، به عنوان یک مسئله NP-کامل توسط جانسون اثبات شده[۹, ۱۸]، که وقتی OCST با چند تابع هدف تعریف شود، پیچیدگی آن به مراتب بیشتر نیز می‌شود.

یک گراف کامل بدون جهت G(V,E) را در نظر بگیرید. در این گراف برای هر جفت از گره‌ها هزینه مکانی $c_{i,j}$ ، متوسط تأخیر پیام $d_{i,j}$ و درخواست ارتباطی $r_{u,v}$ وجود دارد. ماتریس متناظر با هزینه ها و ماتریس متناظر با متوسط تأخیر پیام و ماتریس متناظر با درخواست ها به ترتیب C ، D و R در نظر گرفته می‌شود. بسته به پیاده‌سازی این ماتریس‌ها را می‌توان متقارن و یا پائین مثلثی در نظر گرفت. بهینه-سازی OCST با چند تابع هدف، بدنبال درخت پوشایی است که هزینه‌ی انتقالی و متوسط تأخیر پیام را کمینه می‌کند.

هزینه انتقال مکانی برای هر جفت گره u وv در یک درخت پوشا با توجه به ماتریس درخواست ارتباطی گراف R بدست می‌آید. برای هر جفت گره فقط یک مسیر در درخت پوشا وجود دارد. فاصله‌ی مکانی مسیر مجموع طول یالهای متعلق به مسیر است. هزینه انتقال مکانی هر جفت از گره های u وv، برابر حاصلضرب $r_{u,v}$ در فاصله مکانی مسیر می‌باشد. با مجموع گرفتن روی همه $\binom{n}{2}$ جفت از گره‌ها، هزینه انتقال-مکانی درخت پوشای داده شده بدست می‌آید. بنابراین تابع هدف اول مطابق زیر تعریف می‌شود:

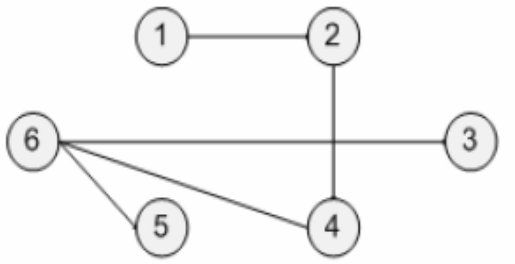
$$\min_{T\in\Im}\left[\sum_{u,v\in V}\left(r_{u,v}\cdot\sum_{(i,j)\in p_{u,v}(T)}c_{i,j}\right)\right] \qquad (۱)$$

توضیح اینکه $\Im$ مجموعه‌ی n^{n-2} درخت بر چسب‌دار روی n گره می‌باشد [۱۹] و $P_{u,v}(T)$ مسیر یکتا بین u وv در درخت پوشای T می‌باشد.

تابع هدف دوم، متوسط تأخیر پیام می‌باشد. متوسط تأخیر مسیر را مجموع تأخیر یال‌های تشکیل دهنده مسیر می‌نامند. متوسط تأخیر پیام هر جفت از گره ها u وv، برابر حاصلضرب $r_{u,v}$ در متوسط تأخیر مسیر می باشد. با مجموع گرفتن روی همه $\binom{n}{2}$ جفت از گره‌ها، متوسط تأخیر پیام درخت پوشای داده شده بدست می‌آید. بنابراین تابع هدف دوم مطابق زیر تعریف شد:

$$\min_{T\in\Im}\left[\sum_{u,v\in V}\left(r_{u,v}\cdot\sum_{(i,j)\in p_{u,v}(T)}d_{i,j}\right)\right] \qquad (۲)$$

برای مثال با در نظر گرفن شبکه‌ای با شش گره که ماتریس‌های C، D و R آن در زیر آمده است توابع هدف برای درخت پوشای شکل (۱) محاسبه شده است.



شکل (۱): یک درخت پوشای نمونه.

$$R=\begin{bmatrix}0&5&13&12&8&9\\0&0&7&4&2&6\\0&0&0&3&10&15\\0&0&0&0&11&7\\0&0&0&0&0&12\\0&0&0&0&0&0\end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 3 & 6 & 5 & 9 & 7 \\ 0 & 0 & 3 & 2 & 4 & 8 \\ 0 & 0 & 0 & 3 & 7 & 2 \\ 0 & 0 & 0 & 0 & 9 & 2 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 5 & 4 & 1 & 1 & 5 \\ 0 & 0 & 7 & 5 & 2 & 2 \\ 0 & 0 & 0 & 1 & 6 & 3 \\ 0 & 0 & 0 & 0 & 7 & 4 \\ 0 & 0 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\cos t(T) = r_{12}(c_{12}) + r_{13}(c_{12}+c_{24}+c_{46}+c_{63}) + r_{14}(c_{12}+c_{24}) + \ldots + r_{46}(c_{46}) + r_{56}(c_{56})$$
$$= 5(3) + 13(3+2+2+2) + 12(3+2) + \ldots + 7(2) + 12(1)$$
$$= 534$$

$$\text{delay}(T) = r_{12}(d_{12}) + r_{13}(d_{12}+d_{24}+d_{46}+d_{63}) + r_{14}(d_{12}+d_{24}) + \ldots + r_{46}(d_{46}) + r_{56}(d_{56})$$
$$= 5(5) + 13(5+5+4+3) + 12(5+5) + \ldots + 7(4) + 12(4)$$
$$= 1120$$

لازم بذکر است که این مسئله با درخت پوشای کمینه معروف (MST) کاملاً متفاوت است [۷, ۲۰].

۳- بهینه‌سازی با چند تابع‌هدف

در بهینه‌سازی با چندتابع، توابع هدف مختلف و گاه متضاد بایستی به طور همزمان بهینه شوند. یکی از راه حلهای مسائل بهینه‌سازی با چند تابع هدف، ترکیب مقادیر آنها و به دست آوردن یک مقدار برازندگی[1] است. بدین صورت مسئله به بهینه‌سازی تک‌تابعی تبدیل می‌شود. برای استفاده از این روش باید مقادیر وزنها توسط کاربر انتخاب شود. روش دیگر بدست آوردن مجموعه ای از جوابهاست که بتواند حداکثر تعداد ممکن از توابع هدف را بهینه سازد. این مجموعه جوابها را بهینه پارتو[2] می‌نامند و با استفاده از رابطه برتری[3] میان جوابهای مختلف این مجموعه حاصل می‌شود. اگر در تمامی فضای جستجو مجموعه جوابهای مسلط نشدنی[4] را بیابیم، آنگاه مجموعه جوابهای بهینه پارتو تشکیل شده است. در این گونه مسایل نمی‌توان میان دو جواب مختلف از مجموعه بهینه پارتو، یکی را بر دیگری برتری داد. پس الگوریتم سعی در رسیدن به تعداد بیشتری از جوابهای مختلف بهینه پارتو خواهد داشت. از مباحث ارائه شده دو اصل پایه‌ای برای بهینه‌سازی با چند تابع هدف وجود دارد: هدایت مسیر جستجو در جهت رسیدن به منحنی جوابهای بهینه پارتو و حفظ و تولید جوابهای بهینه در طول جمعیت جوابها. مراجع [۵, ۲۱] نمومه‌هایی هستند که به مفاهیم و روشهای مختلف بهینه‌سازی با چندتابع هدف پرداخته‌اند.

۴- الگوریتم ژنتیک مبتنی بر درخت پوشا

همان طور که مطرح شد، نحوه‌ی نمایش فضای جواب مسئله از مهمترین قسمت‌های الگوریتم ژنتیک می‌باشد و شامل مرحله‌ی نمایش کروموزوم و مقدار اولیه می‌شود. نمایش کروموزوم در الگوریتم ژنتیک یک نوع ساختار داده‌ای است که جواب‌های نامزد مسئله را به‌صورت کدشده نشان می‌دهد. معمولاً مسائل مختلف ساختارهای داده‌ای مختلفی نیز دارند. در این مقاله ساختار داده‌ای درخت پوشایی که از ایده عدد Prüfer استفاده می‌کند به خدمت گرفته شده است.

یکی از قضیه‌های کلاسیک در شمارش گرافیکی[5]، قضیه کالیله[6] است[۲۲]. بر اساس این قضیه k^{k-2} درخت برچسب دار مجزا روی یک گراف کامل با k گره وجود دارد. Prüfer یک اثبات سودمند از قضیه کالیله با بنا کردن یک تناظر یک‌به‌یک بین این درختها و مجموعه رشته های k-2 رقمی مهیا کرد. به این معنا که ما می‌توانیم با دستکاری فقط k-2 رقم، به طوریکه هر رقم یک عدد بین ۱ و k باشد، یک ارائه یکتا از درخت داشته باشیم[۶]. این دستکاری معمولاً به عنوان عدد Prüfer شناخته می شود. برای هر درخت همیشه حداقل دو گره برگ وجود دارد[۱۴]. با توجه به این قضیه، ما می توانیم به سادگی این کدینگ را مطابق زیر تبیین کنیم

۴-۱ الگوریتم کد گذاری

۱. فرض می کنیم گره i کوچکترین شماره گره‌ی برگ در درخت برچسب دار T باشد.

۲. شماره گره j ، تنها گره‌ای از درخت که به برگ i متصل است، به طرف راست کد اضافه می‌شود. کدینگ از طرف چپ به راست ساخته و خوانده می شود.

۳. گره i و همچنین یال i به j را حذف می کنیم. بنابراین یک درخت با k-1 گره باقی می‌ماند..

۴. مراحل بالا را تکرار می کنیم تا یک یال باقی بماند.

همچنین این امکان وجود دارد که از یک عدد Prüfer یک درخت یکتا تولید کنیم:

۴-۲ الگوریتم کد گشایی

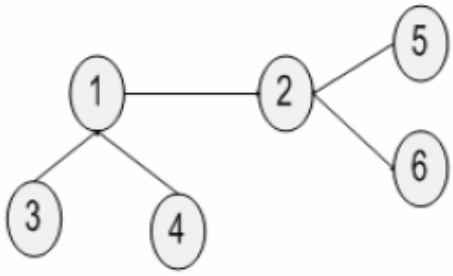
۱. p را آرایه عددی Prüfer اصلی در نظر می‌گیریم و $\overline{p}$ را مجموعه همه گره هایی که در p وجود ندارد قرار می دهیم.

---

[1] Fitness Value
[2] Pareto Optimal Set
[3] Domination
[4] Non-dominated solution
[5] Graphical enumeration
[6] Calyley's theorem

۲. گره i را که کوچکترین شماره در $\overline{p}$ است در نظر می‌گیریم. چپ-ترین رقم p را j می‌نامیم. حال یال(i , j) را در درخت متناظر اضافه می‌کنیم و i را از $\overline{p}$ و j را از p حذف می کنیم.
۳. اگر j در جای دیگر p موجود نبود آن‌را به $\overline{p}$ اضافه می‌کنیم. مراحله دو را تکرار می‌کنیم تا هیچ رقمی در p باقی نماند.
۴. زمانی‌که هیچ رقمی درp باقی نماند دقیقاً دو گره s و r در $\overline{p}$ قابل انتخاب موجود می باشد. بنابراین یال (r , s) را به درخت اضافه می‌کنیم و درخت با k-1 یال را شکل می‌دهیم.

با یک مثال این نوع کدینگ نشان داده می‌شود. یک عدد Prüfer متناظر با یک درخت پوشا روی یک گراف کامل با ۶ گره را در نظر بگیرید.(شکل (۲))



عدد Prüfer :

1	1	2	2

شکل (۲): نمونه‌ای از یک درخت پوشا و عدد Prüfer متناظر با آن.

ساختن عدد Prüfer مطابق زیر توضیح داده می شود:
کوچکترین گره برگ را انتخاب کرده (گره۳) ،چون گره ۱ مجاور گره۳ در درخت است، عدد ۱ را به عنوان اولین رقم Prüfer در نظر می-گیریم و سپس گره ۳ و یال مربوطه (۳و۱) را حذف می‌کنیم. این فرایند را روی زیر درخت به دست آمده تکرار می کنیم تا تنها یال (۶و۲) باقی بماند و در آخر، عدد Prüfer این درخت با چهار رقم به دست می‌آید. و بر عکس برای عدد Prüfer (۱,۱,۲,۲) = p، گره های ۳,۴,۵,۶ قابل انتخاب و$\overline{p}$ = {۳,۴,۵,۶} را تشکیل می‌دهد.. گره ۳ گره قابل انتخاب با کوچکترین برچسب می باشد و گره ۱ چپ‌ترین رقم p می-باشد. بنابراین یالِ (۳و۱) به درخت اضافه می شود وگره ی ۳ از $\overline{p}$ حذف و همچنین رقم ۱ از ابتدای p حذف می شود. (۲,۲, ۱)= p . گره ۴ حالا گره قابل انتخاب با کوچکترین برچسب است لذا گره (۴و۱) نیز به درخت اضافه می‌شود و ۴ از $\overline{p}$ حذف و ۱ نیز از ابتدای p حذف می شود؛ حال چون ۱ دیگر در p وجود ندارد آن را به $\overline{p}$ اضافه می کنیم: {۱,۵,۶} = $\overline{p}$, (۲,۲)= p . این روند ادامه پیدا می‌کند و اینک گره ۱، گره قابل انتخاب از $\overline{p}$ با کوچکترین برچسب می‌باشد ولذا یال(۲و۱) به درخت اضافه می شود. ادامه می دهیم تا p =() و {۲,۶} = $\overline{p}$ شود. در انتها یال (۶و۲) را به عنوان آخرین یال به درخت اضافه می‌کنیم و فرایند به اتمام می‌رسد و درخت پوشای متناظر به دست می‌آید.

کدینگ عدد Prüfer نتنها توانایی ارائه یکتا و متناظر همه درخت‌های پوشای ممکن را دارد ، همچنین به طور ضمنی اطلاعاتی در مورد درجه گره‌ها هم می‌دهد به طوری که هر گره با درجه d دقیقاً d-1 بار در کدینگ آورده می‌شود. بنابراین برای تولید اولیه کروموزمها به صورت تصادفی k-2 رقم در محدوده [1,k] تولید می‌کنیم.

همچنین تائید برتری انتخاب عدد Prüfer به عنوان نمایش ژنتیکی درخت های پوشا در مقاله‌ی[۶]، آمده است.

پس از نمایش هر کروموزوم و مقادیر اولیه، وارد حلقه‌ی تکراری الگوریتم ژیتیک می‌شویم و نیاز به ارزیابی هر کروموزوم می‌باشد که منطبق بر روش‌های بهینه‌سازی چند تابع‌هدفی صورت می‌پذیرد و بر اساس مفهوم پارتو ارزیابی می‌شود، که در اینجا برای هر کروموزوم دو مقدار ارزیاب از دو تابع هدف محاسبه می‌شود.

در ادامه در مرحله‌ی انتخاب نیز، از روش ماندگاری[1] استفاده شده‌است. به این صورت که در این روش با در نظر گرفتن جمعیت جانبی[2] در کنار جمعیت جواب‌های مسئله، آنهایی که در طول نسل جواب‌ها، بهینه هستند را نگهداری می‌کند. بدین ترتیب احتمال از بین رفتن اطلاعات بر اثر عملگرهای ژنتیکی از بین می‌رود. ماندگاری به اجرای بهتر الگوریتم‌های ژنتیک کمک می‌کند، ولی ممکن است باعث همگرایی زودرس شود. در اینجا دو الگوریتم SPEA-IIو NSGA-II، از روش-های بهینه‌سازی تکاملی که هم از مفهوم پارتو جهت بهینه‌سازی استفاده می‌کند، هم روش ماندگاری در آن‌ها به کار گرفته‌شده است، به خدمت گرفته‌شده است.

در آخرین مرحله‌ی حلقه‌ی تکرار به اعمال جهش و ترکیب می‌رسیم. در این مسئله از جهش تک بیتی بر روی رشته‌ی بیتی استفاده شده‌است و در مورد ترکیب نیز، ترکیب تک نقطه‌ای بر روی رشته‌ی عدد Prüfer ،انجام شده‌است تا بین مسیرها شکست صورت بگیرد و شکست و ترکیب منطبق بر حالت واقعی درخت‌ها انجام شود

۵- ابزار PISA

در این مقاله، یک واسط به نام PISA معرفی شده‌است که این اجازه را به ما می‌دهد تا دو قسمت مربوط به مسئله و مربوط به روش بهینه-سازی را از هم جدا کنیم. قسمتی که مربوط به معرفی مسئله است را با عملگرهای تغییر[3] می‌شناسیم و به صورت یک بخش مجتمع درنظر گرفته می‌شود و قسمت مربوط به روش بهینه‌سازی عملگرهای انتخاب[4]

[1] Elitism
[2] External Set
[3] Variation operators
[4] Selection operators

نامیده شده‌است که هر دو قسمت به صورت برنامه‌های مستقل و مجزا پیاده‌سازی شده‌اند و می‌توانند به صورت بسته‌های آماده برای استفاده مهیا شوند و همچنین قابلیت ترکیب به صورت دلخواه را نیز دارا می‌باشد. این ممکن می‌سازد تا پیمانه‌های انتخاب را به صورت مستقل از معرفی مشخص و پیاده کنیم که این از اساس الگوریتم‌های بهینه‌سازیِ چندتابعی مدرن است. از طرف دیگر عملگرهای تغییر هم مجبورند به صورت تک‌پیمانه‌ی مستقل منطبق بر مسئله‌ی بهینه‌سازی تعریف شوند که به توصیف سفارشی مسئله کمک می‌کند. ضمناً این واسط امکان اجرای همزمان چند الگوریتم برای حل چند مسئله مختلف را دارد و همچنین معیارهای مقایسه‌ی حل این مسائل با روش‌های مختلف را هم فراهم کرده‌است[۲۳, ۲۴].

به مسائل بهینه‌سازی با چند تابع هدف، از دو جهت عمده نگاه می‌شود. از یک جهت طرف کاربرد، جایی که مهندسین با مسائل مشکل دنیای واقعی روبرو می‌شوند، بنابراین تمایل دارند روش‌های قدرتمند بهینه‌سازی را به خدمت بگیرند. از جهت دیگر، طرف الگوریتم، جایی که محققین برآنند تا الگوریتم بهینه‌سازیی را طراحی کنند که بتواند برای محدوده‌ی گسترده‌ای از مسائل با موفقیت مورد استفاده قرار بگیرد.

چون روش‌های بهینه‌سازی مدرن به طور فزاینده‌ای پیچیده می‌شوند، کار بر روی هر دو این قسمت‌ها نیاز به تلاش برنامه نویسی قابل توجهی دارد. کم و بیش کدهایی برای روش‌های مختلف بهینه‌سازی و همچنین مسائل آزمایشی مختلف موجود است ولی استفاده از این پیاده سازی‌ها محدود به یک زبان برنامه نویسی یا یک سکوی خاص است. یک توافق کامل روی کد نیاز است تا آن را با کار خودمان یکی کنیم. PISA این امکان را فراهم می‌آورد تا پیاده‌سازیمان را به دو قسمت کاربرد و بهینه‌ساز تقسیم کنیم که مطلوب دو گروه مذکور می‌باشد. یک جداسازی ایده‌آل پیمانه‌های آماده برای استفاده روی هر دو قسمت ذکر شده را مهیا می‌کند و این پیمانه‌ها روی هر قسمت، می‌توانند به صورت آزادانه ترکیب شوند.

برای مهندسین کاربردی مطلوب این است که پیاده‌سازی مسئله‌ی بهینه‌سازیشان را با یک بهینه‌گر موجود، جفت کنند و مسئله‌ی خودشان را حل کنند. هر چند این حالت مشخصاً ممکن نیست که یک بهینه‌گر عمومی موجود باشد تا برای همه‌ی مسائل خوب کار کند، زیرا بسیاری از قسمت‌های یک روش بهینه‌سازی، خیلی مبتنی بر مسئله است.

با مطالعه‌ی بسیاری از مسائل کاربردی واقعی، متوجه می‌شویم که ارائه و معرفی جواب‌های نامزد، کاملاً مبتنی بر مسئله می‌باشد. ملاحظه می‌شود که به طور مثال مسائل بهینه‌سازیِ گسسته که درگیر معرفی‌های پیچیده‌ی شبکه‌ای و گرافی هستند، ترکیب شده از متغیرهای پیوسته و مکانیزم‌های اصلاح خاص هستند. بعلاوه ساختارهای مجاور نیز توسط عملگرهای تغییر، وادار به تحریک می‌شوند – چه به صورت صریح مثل شبیه‌سازی سرد شدن آهسته یا جستجوی Tabu و چه به صورت ضمنی مثل الگوریتم‌های تکاملی- و قویاً روی موفقیت بهینه‌سازی تأثیر می‌گذارند. واضح است مواردی وجود دارند که ارائه‌ی استاندارد مانند رشته‌های باینری و عملگرهای متناسب آن کافی می‌باشند، برای این موقعیت‌ها کتابخانه‌های استاندارد در دسترس می‌باشند تا برنامه‌نویسی را ساده کنند.

خلاصه اینکه، این وظیفه‌ی مهندس کاربردی است تا ارائه‌ی مناسب و ساختارهای مجاور را تعریف کند. بالعکس، بیشتر بهینه‌گرها در زمینه‌ی چند تابع‌هدفی، با عملگرهای انتخاب کار می‌کنند، که فقط مبتنی بر مقدار ارزیاب جواب‌های نامزد هستند و بنابراین مستقل از مسئله می‌باشند. PISA این امکان را می‌دهد تا مکانیزم انتخاب در روش بهینه‌سازی را از قسمت مخصوص مسئله جدا کنیم.

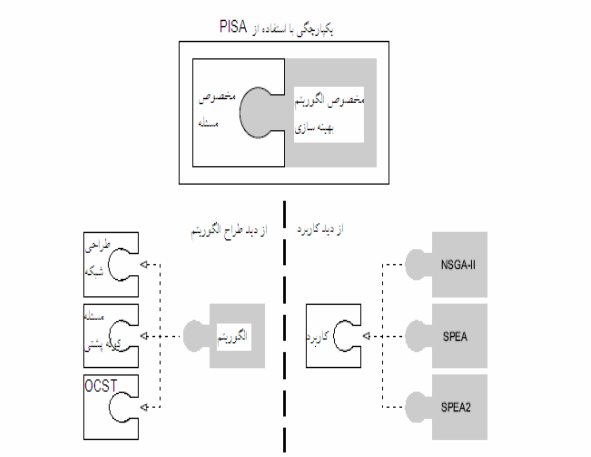
در اکثر تحقیقات انجام شده در زمینه‌ی بهینه‌سازی با چند تابع‌هدف، بر روی مکانیزم‌های انتخاب تمرکز شده است و این الگوریتم‌ها خیلی پیچیده شده‌اند.

در اینجا به رویکردی در این زمینه جهت پی‌بردن به چنین جداسازی اشاره شده که در شکل(۳) به تصویر کشیده شده‌است. قسمت مستقل از مسئله، شامل روتین‌های انتخاب است. در صورتی که قسمت مخصوص کاربرد، ارائه‌ی جواب‌ها و تولید جواب‌های جدید و همچنین محاسبه‌ی مقدار تابع‌هدف را کپسوله می‌کند.

چون هر دو قسمت با برنامه‌های مجزا که از طریق واسطِ بر پایه‌ی متن با هم ارتباط برقرار می‌کنند، مشخص شده‌است، این رویکرد بیشترین استقلال را نسبت به زبان برنامه‌نویسی و سکوی محاسبه (سیستم عامل) دارا می‌باشد. حتی این امکان است تا از فایل‌های پیش‌کامپایل شده و قابل اجرا استفاده کنیم که سربارهای پیاده‌سازی را نیز کمینه می‌کند و از مشکلاتِ ایرادهای پیاده‌سازی نیز جلوگیری می‌کند.

در نتیجه، مهندس کاربردی می‌تواند به سادگی روش انتخاب را عوض و متغییرهای متفاوت را امتحان کند؛ در حالی که یک طراح الگوریتم نیز این مجال را پیدا کند تا روش انتخاب خود را روی مسائل متنوع به سادگی امتحان کند، بدون اینکه نیاز به زحمات برنامه‌نویسی اضافی داشته باشد.

مطمئناً این مفهوم بدین منظور نیست که جایگزین کتابخانه‌های برنامه‌نویسی شود بلکه یک رویکردِ متممی است که این اجازه را می‌دهد تا مجموعه‌هایی از روش‌های انتخاب و کاربردها شامل عملگرهای تغییر را بسازیم که همه‌ی آنها آزادانه قابلیت ترکیب در طرفین این واسط را دارند.



شکل (۳): مفهوم زیربنایی PISA را نشان می‌دهد. کاربردها در سمت چپ و روش‌های انتخاب چند تابع‌هدفی در سمت راست، فقط مثال-هایی هستند و می‌توانند به طور دلخواه جایگزین شوند.

هدف از PISA ، طراحی یک چهارچوب استاندارد، قابل گسترش و ساده برای استفاده، برای پیاده‌سازی الگوریتم‌های بهینه‌سازی چند تابعی می‌باشد. در زیر برخی از اهداف طراحی، که انگیزه‌ی اصلی طراحان PISA می‌باشد آورده شده‌است:

- جداسازی قسمت‌های مهم.
- سربار کم.
- سادگی و انعطاف پذیری.
- قابل جابجایی و مستقل از سکو بودن.
- اعتماد پذیری و صحت.

با در نظر گرفتن این اهداف طراحی، توسعه‌ی یک چهارچوب برنامه-نویسی، خودش یک مسئله‌ی چند تابع‌هدفی می‌شود و این غیر ممکن است که به حداکثر رضایت در همه‌ی جنبه‌ها برسیم. بنابراین یک جواب توافقی، مطلوب می‌باشد. به طور مثال در اینجا با تمرکز بر روی سادگی و سربار کم، کاهش انعطاف پذیری را خواهیم پذیرفت. انگیزه‌ی اصلی موجود در پشت این وضعیت این است که در واقع این سیستم تنها در صورتی توسط بسیاری از مردم مورد به کار گرفته می‌شود که ساده باشد و نیاز به کار برنامه‌نویسی بیش از حد نیز نداشته باشد[۲۳].

۶- آزمایش‌ها و نتایج

در ارتباط با مسئله‌ی OCST ، مثال‌های آزمایشی بسیاری موجود می-باشد که توسط محققان مختلف طرح و ارائه شده‌است. در زیر به طور خلاصه به برخی از آن‌ها اشاره می‌شود:

پالمر[1] در مقاله‌اش مسئله‌ی OCST با ۶ (Palm6)، ۱۲ (Palm12)، ۲۴ (Palm24)، ۴۸ و ۹۸ گره را حل و بررسی کرده‌است. در این مسئله‌ها که OCST به صورت تک تابع هدفی تعریف شده، درخواست ترافیک بین گره‌ها رابطه‌ی معکوس با فاصله‌ی بین گره‌ها دارد. گره‌ها شهرهای ایالات متحده مشخص شده‌اند و فاصله‌ی بین گره‌ها از بانک اطلاعات ترافیک واقعی بین‌شهری تعیین شده‌است. برای داشتن اطلاعات دقیق ماتریس‌های درخواست و فاصله در مورد مسئله‌های با ۶، ۱۲، ۲۴و... گره می‌توان به پالمر ۱۹۹۴ [۱۳]، رجوع کرد.

بِری[2] نیز سه مثال مختلف از مسئله OCST را حل کرده‌است که مسئله با اندازه‌های ۶گره(Berry6) و دوتا ۳۵گره (Berry35 و Berry35u) ارائه شده‌است. هزینه جواب بهینه برای Berry6، ۵۳۴ و برای Berry35، ۱۶،۹۱۵ بدست آمده‌است[۱۰]. ماتریس درخواست ترافیک و فاصله در سایت خودش آورده شده‌است[3]. هر دو این مسائل در مقالات بسیاری مورد بحث قرار گرفته‌است[۷].

رِیدل[4]، در سال ۲۰۰۱، یکسری مثال‌هایی را ارائه داده‌است که مسائل با اندازه‌های بین ۱۰ تا ۱۰۰ گره را شامل می‌شود. وزن‌های فاصله‌ایی و درخواست ترافیک در آن‌ها به صورت تصادفی تولید می‌شوند و به صورت یکنواخت روی بازه‌ی [۰،۱۰۰] توزیع می‌شوند[۲۵, ۲۶].

در این مقاله نیز برای مسئله‌ی OCST با چند تابع هدفی که معرفی شد، مطابق دسته مسائل مطرح شده‌ی اخیر(ریدل)، ماترس‌های درخواست ترافیک، هزینه‌ی مکانی و متوسط تأخیر پیام به صورت تصادفی تولید شده‌است. مسئله در اندازه‌های ۸، ۱۶و ۳۲ گره تعریف و مقادیر هزینه مکانی و متوسط تأخیر پیام به صورت یکنواخت روی بازه-ی [۱،۱۰] و مقادیر درخواست ترافیک شبکه روی بازه‌ی [۱،۲۰] در نظر گرفته شده‌است. در ادامه مثال‌های مختلفی آورده شده‌است که نشان می‌دهد، ارائه‌ی ژنتیکی مطرح شده و همچنین تکنیک‌های چند تابعی آورده شده در این مقاله مؤثر واقع شده‌است و می‌تواند در تعداد نسل‌های نسبتاً خوبی ما را به جواب‌های بهینه نزدیک کند.

در اینجا سه اجرای مختلف از مسئله‌ی با ۸ گره آورده شده‌است و در آنها به مقایسه روش‌های مختلف حل مسائل چندتابعی و مقایسه‌ی نزدیکی جواب‌ها به جواب‌های واقعیِ بدست آمده از روی روشهای الگوریتمیک درخت‌های پوشا مانند نسخه‌ای از الگوریتم پریم (mo-Prim) انجام گرفته است. در الگوریتم mo-Prim با بررسی همه‌ی جواب‌ها ممکن و ارزیابی آن‌ها به صورت چند تابع‌هدفی، به جواب‌های واقعی بهینه پارتو می‌رسیم. در این مثال‌ها از روی مجموعه‌ی کاملِ جواب‌های بدست آمده، جواب‌های بهینه‌ی پارتو استخراج می‌شود و بقیه‌ی جواب‌های مغلوب حذف می‌شوند، و مقایسه بین این جواب‌ها

[1] Palmer (1994)

[2] Berry (1995)

[3] http://www.cse.rmit.edu.au/~rdslw/research.html

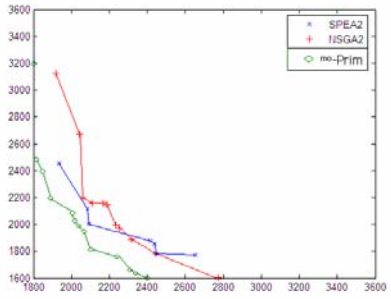
[4] Raidl (2001)

صورت می‌پذیرد(شکل(۳)). در جدول(۱) پارامترهای اصلی الگوریتم‌های به کار گرفته‌شده، آورده شده‌است.

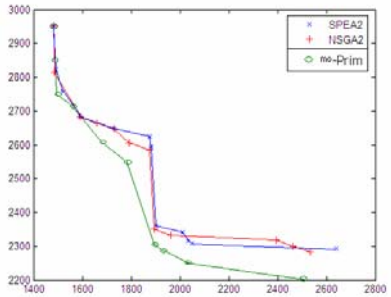
در مورد بقیه اندازه‌های شبکه با تعداد گره‌های بالاتر به آوردن یک مثال از هر اندازه اکتفا می-کنیم و مقایسه بین روش‌های حل چند تابع‌هدفی را خواهیم داشت(شکل‌های(۴)و(۵)). زیرا که در مورد مسائل با تعداد گره‌های بالاتر با توجه به رشد نمایی فضای مسئله، نمی‌توان به راحتی از روش‌های عددی و الگوریتمیک مشابه استفاده کرد[۲۷].

جدول(۱): پارامتر |P| ، اندازه‌ی جمعت را نشان می‌دهد. پارامتر Arc-size ، ظرفیت نگهداری تعداد جواب‌های مغلوب نشده را مشخص می‌کند(مجموعه‌ی خارجی). lambda و mu نیز به ترتیب اندازه‌ی انتخاب تعداد والدین و فرزند جهت ترکیب می‌باشد و p_c احتمال انجام ترکیب بین کروموزوم‌ها و p_m احتمال انجام جهش در یک کروموزوم را در الگوریتم ژنتیک نشان می‌دهد. تعداد تولید هم تعداد نسل‌های تولید شده را نشان می‌دهد.

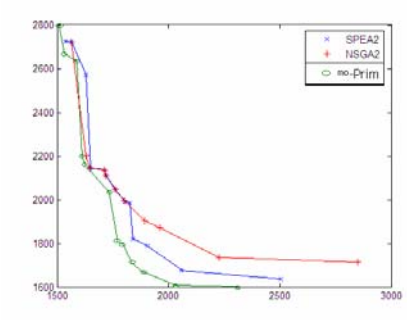
PRÜFER عدد + SPEA-II , NSGA-II						
پارامتر	|P|	Arc-size	lambda=mu	p_c	p_m	تعداد - تولید
۸-الف	۱۰۰	۲۰۰	۵۰	۰,۸	۱	۱۰
۸-ب	۱۰۰	۲۰۰	۵۰	۰,۸	۱	۳۰
۸-ج	۱۰۰	۲۰۰	۵۰	۰,۸	۱	۳۰



شکل(۳) -الف

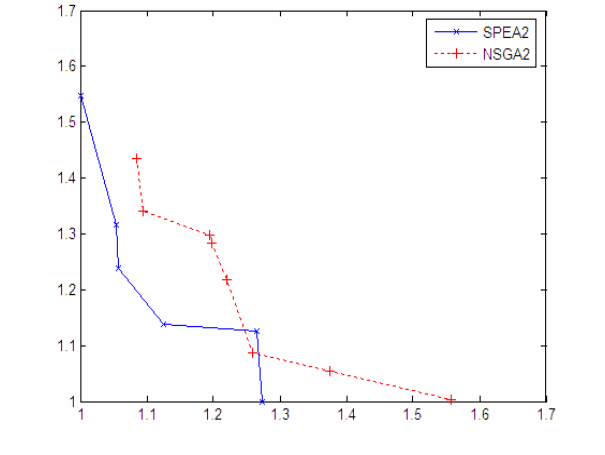


شکل(۳) -ب

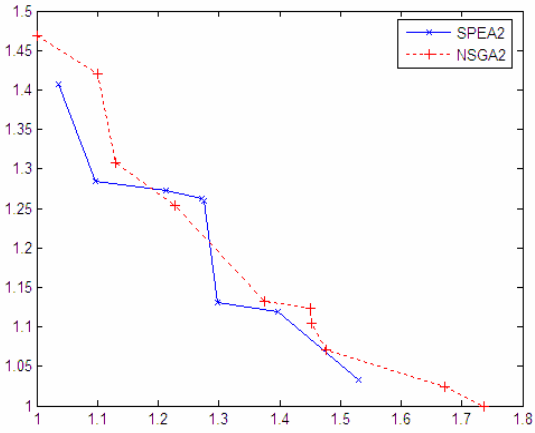


شکل(۳) -ج

شکل(۳) : سه اجرای مختلف روی مسئله‌ی با ۸ گره و مقایسه‌ی جواب‌های بدست آمده از الگوریتم‌های SPEA2 و NSGA2 و همچنین مقایسه‌ی آن‌ها با جواب‌های بهینه‌ی واقعی حاصل از روش‌های عددی



شکل(۴): مقایسه‌ی جواب‌های بهینه‌ی پارتو با دو الگوریتم SPEA2 و NSGA2 برای مسئله با ۱۶ گره



شکل(۵): مقایسه‌ی جواب‌های بهینه‌ی پارتو با دو الگوریتم SPEA2 و NSGA2 برای مسئله با ۳۲ گره

۷- نتیجه‌گیری

با توجه به آزمایشات انجام شده به این نتیجه‌گیری می‌رسیم که برای حل مسئله‌ی OCST با چند تابع هدف به سادگی می‌توان از بهینه-گرهای چند تابع‌هدفی به جای تبدیل مسئله به فرم‌های دیگر، استفاده کرد و با رسیدن به جواب‌های بهینه یا نزدیک بهینه این مدعا اثبات شد. ضمناً با مقایسه‌ی جواب های بهینه بدست‌آمده از هر دو الگوریتم با جواب‌های بهینه‌ی واقعی در نمونه‌های کوچک به این نتیجه می‌رسیم که اگر چه جواب‌ها در هر دو الگوریتم SPEA-II و NSGA-II در بسیاری از موارد نزدیک به هم بود ولی در مجموع کارایی الگوریتم SPEA-II بهتر ارزیابی می‌شود. البته در کارهای آینده به دنبال مقایسه‌ی دقیق‌تر نمونه‌های با تعداد گره‌ی بالاتر با جواب‌های بهینه‌ی تقریباً واقعی بدست آمده از روش‌های الگوریتمیک هستیم.

۸- مراجع

[1] Elbaum, R. and M. Sidi, Topological design of local-area networks using genetic algorithms, *IEEE/ACM Transactions on Networking*, vol.4, No.5, 1996.

[2] Mitsuo Gen, Kenichi IDA & Jongryul KIM, *A Spanning Tree-Based Genetic Algorithm for Bicriteria Topological Network Design*,1998

[3] Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*, 2nd ed., Springer-Verlag, New York, 1994.

[4] Gen, M. and R. Cheng, Genetic *Algorithms and Engineering Design*, John Wiley & Sons, New York, 1997.

[5] E. Zitzler, Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications, Doctorial thesis ETH NO.13398, Zurich: Swiss federal Institute of Technology (ETH), Aachen, Germany: Shaker Verlag, 1999.

[6] Zhou G. and M. Gen, "Approach to degree-constrained minimum spanning tree problem using genetic algorithm," *Engineering Design & Automation*, vol. 3, no. 2, pp. 157–165, 1997.

[7] Li, Y., & Bouchebaba, Y. (1999). A new genetic algorithm for the optimal communication spanning tree problem. In Fonlupt, C., Hao, J.-K., Lutton, E., Ronald, E., & Schoenauer, M. (Eds.), Proceedings of Artifcial Evolution: Fifth European Conference (pp. 162{173). Berlin: Springer.

[8] Goldberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning. Reading, MA: ddison-Wesley.

[9] Johnson D.S., Lenstra J.K., Rinnooy Kan A.H.G. 1978: *The complexity of the network design problem*. Networks, vol. 8(1978) 279-285

[10] Berry, L. T. M., Murtagh, B. A., & McMahon, G. (1995). Applications of a genetic-based algorithm for optimal design of tree-structured communication networks. In Proceedings of the Regional Teletraffic Engineering Conference of the International Teletraffic Congress (pp.361{370). Pretoria, South Africa.

[11] Abuali F. N., R. L. Wainwright, and D. A. Schoenefeld, "Determinant factorization: A new encoding scheme for spanning trees applied to the probabilistic minimum spanning tree problem," in *Proceedings of the Sixth International Conference on Genetic Algorithms*, Larry J. Eshelman, Ed. 1995, pp. 470–477, Morgan Kaufmann.

[12] Palmer C. C., Kershenbaum A. 1995: An approach to a problem in network design using genetic algorithms. Networks, vol. 26 (1995) 151-163

[13] Palmer C. C. 1994: An approach to a problem in network design using genetic algorithms. PhD Thesis, Polytechnic University, Computer Science Department, Brookly, NewYork.

[14] Tanenbaum, A. S., *Computer Networks*, 3rd ed., Prentice-Hall, NewJersey, 1996.

[15] Skiena, S., *Implementing Discrete Mathematics Combinatorics and Graph Theory with Mathematica*, Addison-Wesley, Reading, MA, 1990.

[16] Hu, T. C. (1974, September). Optimum communication spanning trees. SIAM Journal on Computing, 3 (3), 188-195.

[17] Gen, M. and Y. Z. Li, Spanning tree-based genetic algorithm for bicriteria transportation problem, *Proceedings of Japan-China Joint International Workshops on Information System*, pp.123-134, 1997.

[18] Crecenzi P., Kann V. 1998: A compendium of NP optimization problems. aug. 1998.

[19] Gibbons A. 1995: Algorithmic graph theory. Cambridge University Press, New York.

[20] Cormen, T. H., C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, The MIT Press, 1990.

[21] K. Deb, Evolutionary algorithms for multi-criterion optimization in engineering design, In Proceedings of Evolutionary Algorithms in Engineering and Computer Science (EUROGEN'99), 1999.

[22] Cayley, A. (1889). A theorem on trees. Quarterly Journal of Mathematics, 23 , 376-378.

[23] Stefan Bleuler, Marco Laumanns, Lothar Thiele, and Eckart Zitzler. PISA { A Platform and Programming Language Independent Interface for Search Algorithms. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization (EMO 2003)*, volume 2632/2003 of *LNCS*, pages 494-508. Springer-Verlag Heidelberg, 2003.

[24] Stefan Bleuler, Marco Laumanns, Lothar Thiele, and Eckart Zitzler. The PISA Homepage. http://www.tik.ee.ethz.ch/pisa/, 2003.

[25] Raidl, G. R. (2001, February). Various instances of optimal communication spanning tree problems. personal communciation.

[26] Franz Rothlauf, Juurgen Gerstacker, and Armin Heinz, On the Optimal Communication Spanning Tree Problem l, Working Paper 10/2003. May 2003.

[27] Joshua D. Knowles and David W. Corne, *A Comparison of Encodings and Algorithms for Multiobjective Minimum Spanning Tree Problems*. UK, 2001.